



PHD

Process Comprehension for Interoperable CNC Manufacturing

Zhang, Xianzhi

Award date:
2012

Awarding institution:
University of Bath

[Link to publication](#)

Alternative formats

If you require this document in an alternative format, please contact:
openaccess@bath.ac.uk

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal ?

Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Process comprehension for interoperable CNC manufacturing

Xianzhi Zhang

A thesis submitted for the degree of Doctor of Philosophy

University of Bath

Department of Mechanical Engineering

July 2012

COPYRIGHT

Attention is drawn to the fact that copyright of this thesis rests with its author. A copy of this thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with the author and they must not copy it or use material from it except as permitted by law or with the consent of the author.

This thesis may be made available for consultation within the University Library and may be photocopied or lent to other libraries for the purposes of consultation.

Dedicated to my parents Ruifen He and Jichang Zhang

Abstract

Over the last 40 years manufacturing industry has enjoyed a rapid growth with the support of various computer-aided systems (CAD, CAPP, CAM etc.) known as CAx. Since the first Numerically Controlled (NC) machine appeared in 1952, there have been many advances in CAx resource capabilities. The information integration and interoperability between different manufacturing resources has become an important and popular research area over the last decade.

Computer Numerically Controlled (CNC) machines are an important link in the manufacturing chain and the major contributor to the production capacity of manufacturing industry today. However, most of the research has focused on the information integration of upper systems in the CAD/CAPP/CAM/CNC manufacturing chain, leaving the shopfloor as an isolated information island. In particular, there is limited opportunity to capture and feed shopfloor knowledge back to the upper systems. Furthermore, the part programs for the machines are not exchangeable due to the machine specific postprocessors. Thus there is a further need to consider information interoperability between different CNC machine and other systems.

This research investigates the reverse transformation of the CNC part programmes into higher level of process information, entitled process comprehension, to enable the shopfloor interoperability. A novel framework of universal process comprehension is specified and designed. The framework provides a reverse direction of information flow from the CNC machine to upper CAx systems, enabling the interoperability and recycling of the shopfloor knowledge. A prototype implementation of the framework is realised and utilised to demonstrate the functionalities through three industrially inspired test components.

The major contribution of this research to knowledge is the new vision of the shopfloor interoperability associated with process knowledge capture and reuse. The research shows that process comprehension of part programmes can provide an effective solution to the issues of the shopfloor interoperability and knowledge reuse in manufacturing industries.

Acknowledgements

This work documented in this thesis was carried out during the years 2009-2012 at University of Bath. During this journey, I owe my sincere gratitude to many people for their valuable support to help me reach here.

First and foremost I would like to address my deepest gratitude and thanks to my supervisor, Professor Stephen T. Newman. Thanks for bringing me here, an outstanding university in such a fascinating city, and providing me the opportunity to explore the interesting area of research. I really appreciate you not only for your invaluable help and expertise throughout the course of my PhD, both in work and life, but also for your humour and patience in the “Chinese dentist time” we spent together.

The same gratitude goes to my co-supervisor Dr Aydin Nassehi for guiding me throughout the research work. Your advice, wisdom questioning and inspired discussions with me were pivotal to keep me on the right track along the journey.

I would also like to thank my fellow colleagues for your constant support: Dr Vimal Dhokia, Dr Parag Vichare, Mr Reza Imani Asrai, Mr Abayomi B. O Debode, Mr Mehrdad Safaieh, Mr Zicheng Zhu and Mr Alborz Shokrani.

Thanks to all my dear friends both in UK and China, without you, my PhD life would have been less interesting.

This thesis was co-funded by the Engineering and Physical Sciences Research Council (EPSRC) of UK and the China Scholarship Council (CSC), and I would like to thank both organisations for their generous support.

Finally my heartfelt thanks goes to my family my parents Mrs Ruifen Zhang and Mr Jichang Zhang , my sister Ms Guoling Zhang and my brother Mr Xianhui Zhang, without the support of whom none of this would be possible. I am so grateful for your selfless support during many years of my education. I am sure you will be very pleased to see an end to this work.

Contents

Abstract	i
Acknowledgements.....	ii
List of Abbreviations	viii
List of Figures.....	x
List of Tables	xiii
1. Introduction	1
1.1. Background	1
1.2. Research aims	3
1.3. Layout of the thesis.....	4
2. Research objectives and scope	6
2.1. Introduction.....	6
2.2. Research objectives	6
2.3. Scope of the research	7
2.3.1. Interoperability and knowledge management in CNC machining.....	7
2.3.2. Milling technology in CAX	7
2.3.3. Standards of ISO 14649 and ISO 6983	7
2.3.4. Feature recognition method from part programmes.....	8
2.4. Structure of the thesis.....	8
2.4.1. Review of the state-of-the-art in CAD/CAM/CNC interoperability	8
2.4.2. Review of knowledge management in manufacturing	8
2.4.3. Design of a theoretical framework for universal process comprehension	8
2.4.4. Defining Meta-model of CNC programming languages	9
2.4.5. Investigation of feature/operation recognition from part programmes.....	9
2.4.6. Realisation of a prototype implementation of UPCi.....	9
2.4.7. Validation of the UPCi prototype	9
3. State-of-the-art in interoperable CNC manufacturing.....	10
3.1. Introduction	10
3.2. Evolution of NC technology	10
3.3. CAD/CAM/CNC chain.....	13
3.3.1. Background of CAX technologies	14
3.3.2. Current situation in CAD/CAM/CNC integration	15
3.4. Part programming in relation to CAM/CNC integration	17
3.4.1. Part programming methods: a short history and background.....	18
3.4.2. The ISO 6983 solution	19

3.4.3. Efforts in CAM/CNC integration.....	21
3.5. State-of-the-art of in CAD/CAM/CNC Interoperability	24
3.5.1. STEP-NC enabled manufacturing interoperability.....	25
3.5.2. Other technologies based interoperability	38
3.5.3. Commercial solutions on manufacturing interoperability.....	40
3.6. Critique	41
3.6.1. CNC machines in CAx integration and interoperability.....	41
3.6.2. The requirement for modification of current manufacturing resources	42
4. CAPP and knowledge reuse in CAx manufacturing chain.....	43
4.1. Introduction.....	43
4.2. Computer Aided Process Planning (CAPP)	43
4.2.1. Background.....	43
4.2.2. Scope of process planning	44
4.2.3. Classifications of CAPP	44
4.2.4. Current situation of CAPP	45
4.3. Feature-based technology and feature recognition methods	47
4.3.1. Feature technology	47
4.3.2. Feature recognition methods.....	48
4.4. Knowledge management in manufacturing.....	51
4.4.1. Background of knowledge management	51
4.4.2. State-of-the-art in knowledge management in manufacturing industry	53
4.5. Critique	57
4.5.1. Shopfloor gap	57
4.5.2. Product and process information separation gap	57
4.5.3. Knowledge representation gap	58
5. A framework for universal process comprehension for interoperable CNC manufacturing.....	59
5.1. Introduction.....	59
5.2. Possible solutions	59
5.3. Universal process comprehension	60
5.4. Design considerations for a universal process comprehension.....	62
5.4.1. Minimum modification to current resources.....	62
5.4.2. Universal support architecture	63
5.4.3. Shopfloor knowledge capture ability	63
5.4.4. CAx interoperability.....	63
5.5. Design of the universal process comprehension framework.....	63

5.5.1. Functional view of universal process comprehension	63
5.5.2. Flexible computerised Meta-model of CNC activities.....	64
5.5.3. Reconstruction of process plan in a standardised format.....	66
5.5.4. Functional architecture of universal process comprehension framework.....	67
5.6. Summary	69
6. A meta-model of CNC programming languages.....	70
6.1. Introduction	70
6.2. Comparison of part programming systems on modern CNC machines	70
6.3. A meta-model of CNC programming languages	73
6.4. Modelling NC languages in XML description.....	75
6.5. Summary	82
7. Feature recognition and knowledge capture from part programmes	83
7.1. Introduction	83
7.2. Feature recognition from part programmes for prismatic parts.....	83
7.2.1. Feature recognition from toolpath.....	84
7.2.2. Feature recognition from canned cycles.....	96
7.2.3. Feature identification	97
7.3. Knowledge capture from part programmes.....	99
7.3.1. Milling type operations	101
7.3.2. Drilling type operations.....	105
7.4. Standards compliant representation of resource independent process plan	109
7.5. Summary	112
8. Implementation of a software prototype for the UPCi	113
8.1. Introduction.....	113
8.2. An overview of the UPCi prototype system	113
8.3. Development of the Meta-model of programming dialects	116
8.4. Translation from programming dialects into the Meta-model.....	117
8.5. Feature recognition and knowledge capture	120
8.6. Coordinate system update of STEP-NC data.....	124
8.7. Generation of STEP-NC representation of process plan.....	128
8.8. Prototype integration.....	132
8.9. Summary	133
9. Evaluation of UPCi prototype.....	134
9.1. Introduction	134
9.2. Evaluation method	134

9.3. Test I.....	137
9.3.1. Part I.....	138
9.3.2. Part II.....	142
9.3.3. Part III.....	144
9.4. Test II.....	147
9.4.1. Part I.....	147
9.4.2. Part II.....	154
9.4.3. Part III.....	159
9.5. Results of the evaluations.....	164
10. Discussions, conclusions and future work.....	165
10.1. Introduction.....	165
10.2. Discussions.....	165
10.2.1. Interacting feature recognition from part programmes.....	165
10.2.2. Advantages of the universal process comprehension framework.....	165
10.2.3. Limitations of the universal process comprehension framework.....	166
10.3. Conclusions.....	167
10.4. Contribution to knowledge.....	167
10.5. Future work.....	168
10.5.1. Integration of the resource modelling research.....	168
10.5.2. Extensibility to cover turning operations.....	168
10.5.3. Interfacing with other manufacturing systems.....	168
10.5.4. Towards a universal manufacturing platform.....	168
References.....	170
Appendix A. Publications.....	186
Appendix B. XML specifications of Fanuc and Siemens programming dialects.....	187
B.1. XML specifications of Fanuc 18i.....	187
B.2. XML specifications of Siemens 840D.....	197
Appendix C. Case study results.....	207
C.1. Test components and part programmes.....	207
C.1.1. Part I.....	207
C.1.2. Part II.....	210
C.1.3. Part III.....	219
C.2. Test results: STEP-NC files.....	226
C.2.1. ISO 14649 code generated from Fanuc part programme - Part I.....	226
C.2.2. ISO 14649 code generated from Siemens part programme - Part I.....	230

C.2.3. ISO 14649 code generated from Fanuc part programme- Part II	234
C.2.4. ISO 14649 code generated from Siemens part programme- Part II	243
C.2.5. ISO 14649 code generated from Fanuc part programme- Part III.....	252
C.2.6. ISO 14649 code generated from Siemens part programme- Part III.....	261
C.3. Test results: MPF programmes	270
C.3.1. Siemens part programme generated from STEP-NC file - Part I	270
C.3.2. Siemens part programme generated from STEP-NC file - Part II.....	271
C.3.3. Siemens part programme generated from STEP-NC file - Part III	274

List of Abbreviations

AIM	Application Interpreted Model
ARM	Application Reference Model
AP	Application Protocol
B-rep	Boundary Representation
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
CAPP	Computer Aided Process Planning
CAX	Computer Aided System
CIM	Computer Integrated Manufacturing
CIMS	Computer Integrated Manufacturing System
CNC	Computer Numerically Controlled
DNC	Distributed Numerical Control or Direct Numerical Control
DTD	Document Type Definition
DXF	Drawing Transfer File
EIA	Electronic Industries Association
FAPT	Fanuc Automatic Program Tool
FMS	Flexible Manufacturing System
ICs	Integrated Circuits
IGES	Initial graphics Exchange Standard
IP ³ AC	Integrated Platform for Process Planning and Control
ISO	International Organization for Standardization
JVM	Java Virtual Machine
KQML	Knowledge Query Manipulation Language
NC	Numerically Controlled
NIST	National Institute of Standards and Technology

OWL	Web Ontology Language
PDES	Product Description Exchange for Standard
PDM	Product Data Management
PLC	Programmable Logic Controller
PLM	Product Lifecycles Management
SDAI	Standard Data Access Interface
SFP	Shop Floor Programming system
STEP	STandard for the Exchange of Product data model
STIX	STEP Index Library
UML	Unified Modelling Language
UMP	Universal Manufacturing Platform
UPCi	Universal Process Comprehension interface
VDA-FS	Verband der Automobilindustrie – Flächenschnittstelle (Organisation of the automotive industry - surface translation format)
VM	Virtual Manufacturing
VRML	Virtual Reality Modelling Language
XML	eXtensible Markup Language
XSD	XML Schema Definition

List of Figures

Figure 1.1 - A structured view of the thesis chapters	5
Figure 3.1 - Information lost during the process lifecycle	17
Figure 3.2 - The overall structure of ISO 14649 (Suh <i>et al.</i> 2002a)	26
Figure 3.3 - ST-FeatCAPP architecture (Amaitik and Kiliç 2007)	35
Figure 4.1 - Overview of the MCSG-based hybrid feature recognition algorithm (Gao and Shah 1998)	51
Figure 5.1 - The concept of process comprehension	61
Figure 5.2 - A universal process comprehension interface	62
Figure 5.3 - Overview of universal process comprehension	64
Figure 5.4 - Meta-model of CNC programming languages	66
Figure 5.5 - IDEF0 representation of universal process comprehension framework	68
Figure 5.6 - Interoperable manufacturing enabled by UPCi	69
Figure 6.1 - Process plan in CAM to resource dependent part programmes	71
Figure 6.2 - Meta-model used for universal process comprehension	74
Figure 6.3 - Meta-model used for standardised process comprehension	76
Figure 6.4 - Standardised translation interface using a “dictionary”	77
Figure 6.5 - XML specification for Fanuc 18i controller	78
Figure 6.6 - Part programmes segments for Fanuc 18i controller	79
Figure 6.7 - Translation of part programme to Meta-model	80
Figure 6.8 - XML Schema	81
Figure 7.1 - Feature recognition process	84
Figure 7.2 - Feature identification: cutting area of toolpath	86
Figure 7.3 - Feature identification: a pocket	86
Figure 7.4 - Feature identification: an open pocket	87
Figure 7.5 - Feature identification: a slot	90
Figure 7.6 - Feature identification: a slot	90
Figure 7.7 - Feature identification: a step	91
Figure 7.8 - Feature identification: a planar face	92
Figure 7.9 - Feature definition in STEP-NC (ISO 14649-10 2002)	93
Figure 7.10 - Two types bosses	94
Figure 7.11 - Hole bottom conditions	96
Figure 7.12 - Drilling operation	97
Figure 7.13 - Cutting area of operations of a boss	98
Figure 7.14 - Operations for a pocket	99

Figure 9.11 - Siemens MPF generated from STEP-NC file for Part I.....	151
Figure 9.12 - Three dimensional simulation of Part I.....	152
Figure 9.13 - Part I machined on Siemens machine	152
Figure 9.14 - Test results for Part I.....	153
Figure 9.15 - Part II machined on Fanuc machine.....	154
Figure 9.16 - UPCI during process comprehension of Part II	155
Figure 9.17 - Process comprehension results of Part II.....	156
Figure 9.18 - Part II in the interoperable CAD/CAM system	157
Figure 9.19 - Three dimensional simulation of Part II	157
Figure 9.20 - Part II machined on the Siemens machine	158
Figure 9.21 - Test results for Part II.....	158
Figure 9.22 - Part III machined on Fanuc machine	159
Figure 9.23 - UPCI during process comprehension of Part III.....	160
Figure 9.24 - Process comprehension results of Part III	161
Figure 9.25 - Part III in the interoperable CAD/CAM system.....	162
Figure 9.26 - Three dimensional simulation of Part III.....	162
Figure 9.27 - Part III machined on Siemens machine	163
Figure 9.28 - Test results for Part III	163
Figure 10.1 - Universal manufacturing platform (Newman and Nassehi 2007)	169

List of Tables

Table 3.1 - The Evolution of NC machine tools (adapted from Newman 1982)	13
Table 3.2 - Examples of Heidenhain Programming Language Commands	23
Table 3.3 - Contrasts between different systems	24
Table 6.1 - Comparisons between Fanuc, Siemens and Heidenhain programming dialects	72
Table 6.2 - Meta-model entities of CNC activities.....	75
Table 7.1 - Machine functions and technology mapping between Siemens and STEP-NC	103
Table 7.2 - Mapping between STEP-NC entities and canned cycles of Siemens and Fanuc	106
Table 7.3 - STEP-NC versus part programme in terms of information types	110
Table 8.1 - Process plan sheet	118
Table 8.2 - Recognised operations and features	123
Table 8.3 - STEP-NC process plan	129
Table 9.1 - Test parts.....	137
Table 9.2 - Process plan of Part I	139
Table 9.3 - Comparison between two generated STEP-NC files for Part I	140
Table 9.4 - Comparison between the standard file and the generated file	140
Table 9.5 - Process plan of Part II.....	144
Table 9.6 - Comparison between two generated STEP-NC files for Part II.....	144
Table 9.7 - Process plan of Part III	146
Table 9.8 - Comparison between two generated STEP-NC files for Part III	147

1. Introduction

1.1. Background

Since the first Numerically Controlled (NC) machine was introduced, machine tools have evolved in both their architecture and control as well as the accessory equipment and software (Zhang *et al.* 2006). With the rapid development of computer technology, NC machines were developed into Computer Numerically Controlled (CNC) machines (Toh and Newman 1996). Nowadays, CNC manufacturing is not limited to a standalone machine tool on the shopfloor. Usually it encompasses a huge system consisting of many assisting subsystems and mechanisms. The major subsystems include Computer Aided Design (CAD), Computer Aided Process Planning (CAPP), Computer Aided Manufacture (CAM) and CNC. The integration and interoperability of these systems are becoming increasingly complicated due to the different standards and requirements (Nassehi 2007).

In order to make them work together efficiently and seamlessly, researchers have proposed various methodologies and terms from the early Direct Numerical Control (DNC) to Flexible Manufacturing System (FMS) in the 1980s and Computer Integrated Manufacturing (CIM) in the 1990s to today's reconfigurable manufacturing systems (Dhupia *et al.* 2007; ElMaraghy 2005). These solutions aimed to increase the efficiency and productivity of CNC machining. However, no one provides a satisfying answer to the interoperability problem due to the lack of a standardised and applicable product and control data format, especially for the CNC machines at the shopfloor.

The information flow from CAD/CAPP/CAM systems to the CNC machine is unidirectional and separate for each machine since different programming languages or dialects are used on different machines. Each machine is an information island and it is virtually impossible to efficiently exchange part programmes between different CNC machines. The situation is quite different from the way computer printers work. In the case of printers, they all accept data in a postscript format. Thus it does not matter what kind of operating system is running on a computer and which program is requesting the printing services, different printers can be used to perform the task. The problem of CNC machines using different languages roots from the parallel development of different CNC machines around the world. There was no dominating or standardised data format to

control the machines. This situation is quite harmful to modern manufacturing industry. Since there is no interoperability and flexibility at the shopfloor, there is a need to develop separate part programmes for each machine used to machine the same part. For legacy part programmes, it is impossible to reuse their part programmes on modern CNC machines. If the part programmes were developed by hand or the CAD/CAM data has been lost, the part programme has to be discarded and a new part programme has to be developed from the beginning.

Interoperability or the capability to seamlessly transfer information from one computer system to another without loss of data (Nassehi 2007) is quite essential to the modern manufacturing industry. Interoperability offers flexibility and adaptability. In a global economy with volatile demands, the flexibility and adaptability leading to quick response to new needs and short lead-time to new product development are crucial. Although STEP and STEP-NC can be considered good candidates to resolve the problem of interoperability, the implementation is arduous as a result of the variants of data formats in use based on G&M codes (formally known as ISO 6983) used for programming CNC machine tools (Newman *et al.* 2008).

On the other hand, with the use of modern information systems, a knowledge driven economic environment has been emerging (Manufacture 2004). The effective utilisation of previous knowledge is becoming increasingly important to sustain competitive advantage (Baxter *et al.* 2008). Commercial companies can benefit from knowledge reuse in new product development in terms of ensuring the quality, short lead-time, high value output.

In the NC manufacturing chain, computer aided systems (collectively known as CAx) have been developed with the advancement of computer technology to automate the production process (Xu and He 2004). They assist and prepare processes for actual production. The goal of these tools is to automate the preparation for production with the help of computers. Hence, shopfloor machining is the most important stage of the CAD/CAM/CNC chain. The shopfloor is a knowledge intensive and ever-changing environment with issues such as machine tools being bottlenecks, tool breakage and also machines being broken down. The operators are typically empowered to make necessary changes or improvements to the part programmes according to their experience and the real-time facility status. It also happens when the part programme is not well prepared

such as with inappropriate cutting speeds, feedrates etc. An alternative scenario could be that manufacturing companies have been using and updating the part programmes at the shopfloor for many years and the original programmer(s) is not available anymore. It thus makes the shopfloor part programmes a critical knowledge carrier. Nevertheless, there has been a lack of efficient methods to recycle and reuse the knowledge in the existing part programmes due to the interoperability issues within the shopfloor.

In existing academic research and commercial products, system interoperability and knowledge management is typically addressed in the upper systems including CAD, CAPP, and CAM etc. There is limited work reported on the interoperability and knowledge reuse issues concerning the shopfloor, or CNC machines tools. However, in the author's opinion, the shopfloor is the most important stage of CNC production. All other computer systems are assistant tools to make it happen correctly at the shopfloor. Thus a practical solution to address the shopfloor interoperability and knowledge reuse has become necessary.

In this research, a process comprehension approach has been proposed to tackle the interoperability issues in connection with CNC machines at the shopfloor. Process comprehension here is the appreciation and interpretation of process knowledge contained in various part programmes.

1.2. Research aims

The aims of the research have been identified as:

- (i) to improve shopfloor data and knowledge loss;
- (ii) to reuse legacy process knowledge;
- (iii) to increase the shopfloor flexibility and improve interoperability.

To achieve the aims, the following technological requirements are necessary:

- (i) A through technological understanding of the part programming codes for each CNC machine is required. This is critical so that an accurate understanding of the meaning of each code and the integrity of information transfer between systems can be achieved.

- (ii) A set of sophisticated algorithms needs to be developed to extract the process information from the part programmes and convert the part programmes into high level process information.

1.3. Layout of the thesis

The research documented in this thesis has been organised into 10 cohesive chapters as shown in Figure 1.1. The opening two chapters are introduction, research aims, research objectives and scope. Chapter 3 reviews the research on interoperability. The second review presented in Chapter 4 outlines the research on knowledge management in manufacturing. The research gaps and opportunities have been identified based on the reviews. A theoretical research framework of process comprehension interface for shopfloor interoperability is specified, envisioned and developed in Chapter 5. Two realisation elements of the framework are presented in detail in Chapter 6 and Chapter 7. In Chapter 6, a Meta-model of CNC programming languages is proposed. Various different part programmes can be translated into this model. In Chapter 7, feature recognition methods based on the data of this Meta-model are presented. In the experimental phase, a prototype implementation of process comprehension is realised, demonstrated and evaluated in Chapter 8 and Chapter 9. Finally the conclusions drawn from the research together with areas for potential future research are documented in Chapter 10.

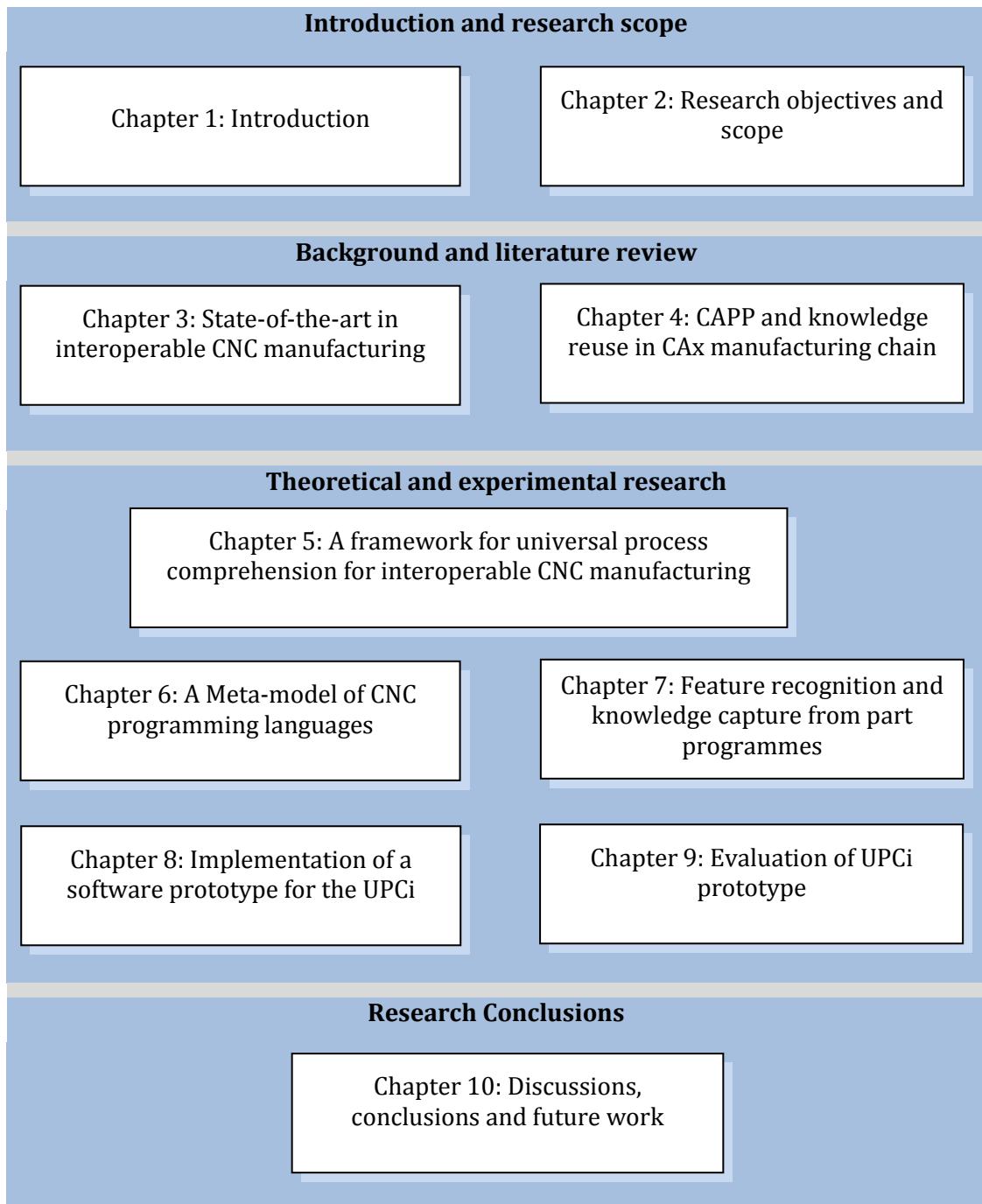


Figure 1.1 - A structured view of the thesis chapters

2. Research objectives and scope

2.1. Introduction

In this chapter, the research aims, objectives and research boundaries have been outlined. The first part of this chapter outlines the research aims of enabling interoperability between the shopfloor and other manufacturing resources and knowledge reuse. The boundaries to develop a process comprehension interface in the context of CNC manufacturing are formalised. A set of objectives to realise the aims are ascertained, which are then used to define the scope of the research and areas of investigation.

2.2. Research objectives

To achieve the research aims, the following objectives have been identified:

- To understand the root cause of why G&M codes can't be reused and Why is it a problem to capture and reuse the process knowledge in part programmes.
- To identify the possible solutions for extracting process knowledge from G&M codes.
- To design a research framework for shopfloor interoperability and knowledge reuse. In this framework, the following sub-objectives are identified:
 - A standardised model to represent various part programming dialects to realise interoperability in a universal manner for different CNC machines.
 - Effective product and process recognition algorithms based on part programmes.
 - A generic data structure to accommodate the process information for sharing with other manufacturing resources.
 - A prototype implementation of the framework to demonstrate the applicability of the framework.

- To evaluate the prototype implementation by utilising test components with appropriate features and state-of-the-art commercial software and hardware tools.

2.3. Scope of the research

This research is conducted in the context of CAD/CAM/CNC manufacturing chain. The main issue in this research spans four areas, which also defines the boundaries of this research. These areas are:

2.3.1. Interoperability and knowledge management in CNC machining

Due to the advancement in computer technology, computer based systems are used everywhere across the manufacturing industry. The major issue of the integration of these systems is the communication between each other. In this research, the focus is laid on the interoperable information (knowledge) sharing of the CNC machining systems. As the investigation of literature review shows, the interoperability between CNC controllers and other manufacturing systems is limited and does not receive much attention. The knowledge management regarding shopfloor needs further investigation. In this research, the shopfloor interoperability has been chosen to be the major focus to facilitate the process knowledge recycling and reuse.

2.3.2. Milling technology in CAx

CNC technology has been used in many different manufacturing processes including both the subtractive and the additive process. According to the literature and avoiding the complexity dealing with multiple manufacturing technologies, prismatic parts and 3-axis milling have been chosen as the main area of interest within the CAD/CAM/CNC domain.

2.3.3. Standards of ISO 14649 and ISO 6983

ISO 6983 (ISO6983-1 1982), more commonly referred to as G&M codes, has been used for more than 50 years without considerable improvements. It is believed that it limited the development of entire manufacturing industry for rudimentary low-bandwidth information transfer capability (Nassehi 2007). From the literature review in section 3.5.1, STEP-NC has been developed as the new data interface for computerised numerical controllers. In this research, ISO 14649 has been chosen as the information standard and

data accommodating mechanism due to it being both comprehensive and concise on the relevant information involved in this research.

These two standards are extensively used in Chapter 5 where a universal process comprehension interface (UPCi) accommodating these two standards has been presented.

2.3.4. Feature recognition method from part programmes

STEP-NC provides a hierarchical data structure to store feature-based manufacturing information while G&M code only contains low-level tool movements and switching instructions of the machine tool. Consequently, a suitable feature recognition method should involve the conversion between G&M code and STEP-NC. According to characteristics of G&M code, a novel feature recognition method is proposed. It is different from the traditional understanding of feature recognition since it is based on part programmes, raw workpiece and cutting tool information, not CAD data.

2.4. Structure of the thesis

In correspondence of the research objectives, the structure of the thesis is listed below.

2.4.1. Review of the state-of-the-art in CAD/CAM/CNC interoperability

Considerable research and development efforts have been paid on the interoperability issue of manufacturing chain. The existing literature has been reviewed and accessed in Chapter 3 to identify the gaps for the current research.

2.4.2. Review of knowledge management in manufacturing

Knowledge management is a huge subject used in a wide range of disciplines. The literature relating manufacturing knowledge management is reviewed in Chapter 4 to highlight the shopfloor ignorance. To highlight the significance of the shopfloor knowledge in existing part programmes, a short review of CAPP is conducted.

2.4.3. Design of a theoretical framework for universal process comprehension

Based on the literature critiques, different possible solutions have been proposed and compared, and a universal process comprehension approach have been selected. A theoretical research framework of process comprehension has been designed in Chapter

5. It is designed to achieve the research aims and forms the skeleton of the thesis. The IDEF0 methodology is used to specify the information models and procedural activities of the framework.

2.4.4. Defining Meta-model of CNC programming languages

In order to standardise the process comprehension activities regarding different programming dialects, a Meta-model of various CNC programming languages has been developed in Chapter 6. It is an abstraction of CNC activities. An approach using XML schemas translating programming dialects into the Meta-model is proposed.

2.4.5. Investigation of feature/operation recognition from part programmes

To recover the process information from part programmes and store it into a standardised format, the feature and operation recognition methods have been investigated. A range of individual and interacting features are covered in the recognition algorithms. The generation of standardised representation of the process plan has been addressed.

2.4.6. Realisation of a prototype implementation of UPCi

Utilising the Meta-model of CNC languages developed in Chapter 6 together with the process comprehension methods presented in Chapter 7, a prototype implementation of universal process comprehension interface has been provided in Chapter 8.

2.4.7. Validation of the UPCi prototype

The prototype implementation has been evaluated through three test components in Chapter 9. The test parts utilised in the experiments have been designed with necessary complexity and industrial significance.

3. State-of-the-art in interoperable CNC manufacturing

3.1. Introduction

Interoperability is an imported concept to manufacturing due to the intensive use of various computer systems in modern manufacturing industries. It describes a sophisticated level of an automatic error-free information sharing mechanism between them. This chapter presents a review of the state-of-art solutions to manufacturing interoperability. To achieve this, the history of NC technology and the integration of the CAD/CAM/CNC chain, collectively as Computer Aid x (CAx) systems, have been reviewed to highlight the trends of interoperability in manufacturing. Various interoperability-enabling methodologies have been reviewed. Critiques of existing literature have been developed to conclude the research gaps and opportunities of this research and serve as a basis for constructing the requirements of this work.

3.2. Evolution of NC technology

There are many definitions of Numerical Control, commonly termed NC. The Electronic Industries Association (EIA) defines it as “a system in which actions are controlled by the direct insertion of numerical data at some point. The system can automatically interpret at least some portion of the data” (Lin 1994). NC can be considered as the natural evolution from the conventional manufacturing process where the skill of the operators is captured in the input medium (*e.g.* punched paper tapes, initially). NC aims to provide precise and efficient manufacturing in a reliable repetitive manner.

NC is one of the most important parts of the whole concept of the industrial automated technology and has now been used for more than 50 years (Tsuji 2003). Today, NC technology is playing an important part to the production capacity of industrial companies (Newman and Nassehi 2007). However, the thought and effort to control a machine by programmed media is nothing new. As early as in 1808, weaving machine used metal cards with holes punched in them to control the pattern of the cloth being produced (Curran and Stenerson 2001). After that, various ingenious activities have contributed to the birth of the first NC machine.

The initial need for development of NC was to machine complicated shapes, such as the curved surfaces in the aircraft industry (Newman 1982). In 1949, the United States Air

Force funded a project in Massachusetts Institute of Technology to build the first NC machine tool and operating software (Lin 1994). Three years later, in 1952 the first NC machine, a Cincinnati Hydrotel Vertical-spindle milling machine with three axis control, emerged. However, the first real operational machine of this kind was not installed until 1957 (Newman 1982).

Basic elements of a NC system include: machine control unit, drive system, machine tool and feedback system. Since the successful introduction of the first NC milling machine, NC technology has undergone many significant enhancements (Zhang *et al.* 2002). In reaction to the advancements in computing technology, the machine control unit has changed tremendously, and these improvements have contributed significantly to development of NC technology. According to the different stages of the development of the machine control unit, NC machine tools can be classified into several “generations” (Newman 1982).

The first generation of NC machine tools was introduced commercially in 1955 (Tsuji 2003), and the control unit was based on valves and analogue hard wired circuitry. Consequently, it is huge, costly and unreliable. At this stage, the control unit was fitted to conventional machines, which led to high rate of wear and inaccuracy. Most of them were point-to-point that positions the tool from one point to another within a coordinate system. Machining, mostly drilling, boring, punching etc., can only take place after positioning is completed (Pusztai and Sava 1983).

The second generation of NC machine tools in 1959 was based on digital circuitry using individual transistors and other discrete components. The machine tools were designed specifically for NC with attention being given to wear and backlash for greater accuracy and reliability.

The third generation was introduced about 1965 and used Integrated Circuits (ICs). ICs are smaller and have better performance, which led to easier maintenance and better utilization of NC. Machine tools were better designed and tool changers and machining centres appeared.

The fourth generation was actually introduced in 1964 and was known as Direct Numerical Control (DNC), but it was around 1970 before it had any impact. Direct Numerical Control shared expensive interpolators across several NC machine tools and

effected reliable transfer of control information (Stute and Nann 1971; Toh and Newman 1996). The centralised computer was primarily used to download part programs to the NC machine, with their use and function being very limited.

The fifth generation emerged in early 1980s named Distributed Numerical Control with the same acronym DNC. From then on NC technology entered into a new soft wired control era named CNC with mini-computers embedded in the machine tools instead of the hard wired techniques. The major difference between Distributed Numerical Control and Direct Numerical Control is the “on-line” fashion of Direct Numerical Control, since the substitution of conventional hard-wired NC with soft-wired CNC allowed the whole part program to be transferred into the memory of the CNC (Toh and Newman 1996). The host computer and the CNC machine tools form a data communication network system. In addition to downloading part programs to CNC machines, the system can also handle several important functions such as line balancing and scheduling, monitoring of machines and control status data, and generation of management information (Lin 1994).

The sixth generation is not as straightforward as previous generations with significant hardware improvement. With development in computing technology, the computers have become more and more integrated with the NC technology and new more advanced systems have appeared such as Flexible Manufacturing System (FMS) (Buzacott and Yao 1986), Integrated Manufacturing System (IMS) (Platts 1995; Lee 1993) and Computer Integrated Manufacturing (CIM) (Anjard 1995), which are more intelligent and integrated. In these systems, different Computer Aided systems, including CAD, CAPP, CAM, are used during the whole process from product design to manufacturing. However, except for the computing power upgrade of the control unit, the CNC machines haven't changed much compared to the last generation, while users seek different ways to make the most use of them.

The latest generation emerged a few years ago in an attempt to achieve changeable functionality and scalable capacity (Dhupia *et al.* 2007). The Reconfigurable Manufacturing System consists of different modules that can be added, removed, modified, or interchanged as needed to respond to the volatile market. “It has the potential to offer a cheaper solution in the long run compared to FMS, as it can increase the life and utility of a manufacturing system.” However, “hardware reconfiguration also requires major changes in the software used to control individual machines, complete

cells, and systems as well as to plan and control the individual processes and production” (ElMaraghy 2005). A summary of characteristics of these generations of NC machine tools is illustrated in Table 3.1.

Table 3.1 - The Evolution of NC machine tools (adapted from Newman 1982)

Generation		Name	Characteristics	Start year
NC	I	Valves and analogue hard wired circuitry	High rate of wear and inaccuracy	1954
	II	Individual transistors and discrete components	Greater accuracy and reliability	1959
	III	Integrated Circuit	Easier maintenance and better utilisation	1965
	IV	Direct Numerical Control	Employing a remote computer to operate both management and system control over several NC machine tools	1970
CNC	V	Distributed Numerical Control	Shopfloor communication network with extension functions	Early 1980s
	VI	FMS / IMS /CIM	CAD/ CAPP/ CAM/ CNC integration	1980's /1990's
	VII	Reconfigurable manufacturing systems	A machining system which can be created by incorporating basic process modules - both hardware and software- that can be rearranged or replaced quickly and reliably (Mehrabi <i>et al.</i> 2000).	2000

3.3. CAD/CAM/CNC chain

The increasing complexity of CNC machining necessitated the software tools to support computer based manufacturing. This section provides an introduction and overview of these computer aided systems to investigate the integration problem between them.

3.3.1. Background of CAx technologies

With the rapid development of NC/CNC technology, the modern CNC machines have got many advanced capabilities of such as multi-axis control, adaptive control, error compensation and multi-process manufacture (Nguyen and Stark 2005). With the versatility of CNC, the programming task becomes increasingly more difficult. For some precision and complex jobs, it is impractical to program at the shopfloor, which makes offline computer aided software tools a necessity for efficient generation and verification of NC code. On the other hand, the involvements of CAx systems simplify the process and enhance the product development efficiency greatly. It enables people to develop new products easily without strong mathematical and production engineering background.

The basic role of a CAD system is a computer-based interactive environment to design, analyse, create or modify the digital model of the product. CAM is another computer solution using the e-design of the product, combining with the information related to the available manufacturing resource, to figure out the suitable schema to machine the part as similar to the design as possible (Xu and He 2004). CAD, CAM and CNC systems formed a CIM system (Brandimarte and Cantamessa 1995).

Sometime a CAPP system is used before the involvement of a CAM system. CAPP is interpreting the design data to decide how to machine the part with what kind of resources. However, in practice a CAPP system is usually embedded in a CAM system to avoid the problem of integration of CAPP and CAM systems. Thus in this thesis, CAx manufacturing chain is equivalent with CAD/CAM/CNC without CAPP. The literature on CAPP is explored in detail in Chapter 4.

In the 1980s and 1990s, the market for CAD and CAM became very competitive. In order to match the advancement of CNCs, each software vendor tried to support the new capabilities of the latest generation of CNC machines to preserve and expand their market share. Furthermore, both CAD/CAM vendors and CNC manufacturers were very protective of every advance they made and employed proprietary standards for the enhancements that they introduced in their new products (Nassehi 2007).

For example, CNC controller manufacturers introduced non-standard G-codes into the ISO 6983 standard to support their new features (additional axes, special canned cycles) resulting in various dialects for different machine controller combinations (Proctor *et al.*

2002). Companies like Mazak even introduced its total solution where both the hardware and software were provided by the same vendor. Each system could be seamlessly integrated with others in the CAx chain. However, it was impossible to accommodate other vendors' systems in the whole process. Meanwhile CAD vendors utilised proprietary formats like DWG and DXF (Autodesk 2012) to store product design information. CAM vendors have to make efforts to provide comprehensive postprocessors for each machine-controller configuration to be able to generate the correct dialect of G-codes for that specific combination (Hardwick and Loffredo 2006).

Generally, the advances in CAx technology resulted in a plethora of languages and proprietary standards (Nassehi 2007). He claimed that the CAx chain has become a set of isolated islands of automation where information links have to be maintained through low-bandwidth information transfers suitable for the lowest denominator.

3.3.2. Current situation in CAD/CAM/CNC integration

Integration is the process of combining software components, hardware components or both into an overall system (IEEE 1990).

In manufacturing, the CAD, CAM and CNC machines, representing product design, process planning and machining, together with other manufacturing resources compose an overall collaborating product manufacturing system. However, product design, process planning and shopfloor machining can be accomplished in different enterprises or individual departments (Jardim-Goncalves *et al.* 2006). Hence there is a need for seamless data exchange so that each party can achieve its goal. The communication and collaboration between CAx systems became a popular research area of CAx integration. Basically there two methods to realise the communication between different systems: utilising interfaces and utilising neutral data formats (Mokhtar and Houshmand 2010). Currently the interface method is widely adopted in practice since there is no universal data format available at the early stage.

There are two kinds of interfaces used in the CAD/CAM/CNC manufacturing chain, the data exchange interface between CAD and CAM systems or between different CAD or CAM systems and the interface for the communication between CAM systems and CNC machines. Since the outputs of most CAM systems are tool path based part programs, it contains low-level tool movements and switching operations. As a result, a 3D design

model from CAD is adequate as the input of a CAM system to produce part programmes. There are many data formats developed for the information exchange between CAD and CAM, which are generally thought to be sufficient now (Xu and He 2004). The most widely accepted formats for data exchange have been the Initial Graphics Exchange Standard (IGES), the Drawing Transfer File (DXF), the Product Description Exchange for Standard (PDES), the Verband der Automobilindustrie – Flächenschnittstelle (VDA-FS) and the STandard for the Exchange of Product data model (STEP) (Kern and Böhn 1997). Another reason that the CAD/CAM interface is acceptable is that there are many commercial packages used in practice, which combines CAD/CAM together.

As for the interface linking the CAM and CNC, the data delivered through it is the control information. It is commonly accepted that this link is so weak to supply the controller with sufficient information to realise an intelligent manufacturing, which is rooted from the long-established standard-ISO 6983. The major problem with this standard is its low-level information representation and loss of geometry data. Actually, there is lot information lost during the product lifecycle. As shown in Figure 3.1, at the level of the CNC machine, most of the information except axis movements is lost. G&M codes just describe the axis movements and machine functions to order the machine to carry out the required motions. The CNC machine is only an obedient executor without knowing what is being machined on the table.

The connection between CAM and CNC is a bottleneck of the CAx chain due to the information loss and low-level information representation of G&M codes. This bottleneck has become the major obstacle for an unimpeded information highway through the CAx chain. Even the same standard (ISO 6983) employed, different CNCs use different dialects. It means for each pair of CAM/CNC a dedicated postprocessor is needed, which is expensive burden on users. To solve the problem, some vendor companies provide their own solutions by developing their own programming languages and software (Mazak 2010; Heidenhain 2009). However, it is impossible to be used world-wide.

The reasons of the interface problems around the CNC manufacturing chain are comprehensive. However, two elements are the major cause of this situation. The simultaneous development of CNC machines around the world is the major cause of various languages used on CNC machines. The standard of programming CNC machines actually came many years later after the first numerical controller machines launched

and the standard is based on the most dominating G&M codes from the various controllers. Another reason is that since the NC machines evolved into CNC machines, the computing technology has undergone a number of fundamental improvements. The shopfloor computing capability has increased enormously, which enables CNC machines on the shopfloor to perform many other tasks not just machining such simulation, inspection etc. The CNC machines have become a standalone system not merely a machine tool. This has encouraged the development of new programming languages or dialects and a new shift such as shopfloor programming (CADF/CAM) systems. It is quite different with rapid prototyping machines. It did not undergo booming development around the world like what happened to NC machines in the early stage. The rapid prototyping machines are not like CNC machines with separate computers embedded at the machine tools since at the shopfloor there is no need to perform complex tasks for production. The data (STL) used to programme rapid prototyping machines is quite adequate for the task and there is no need, at least not now, to develop proprietary languages to control a rapid prototyping machine. In the future, alternatives or additives to STL will be required if more advanced machines such as multi-material machines are to be developed.

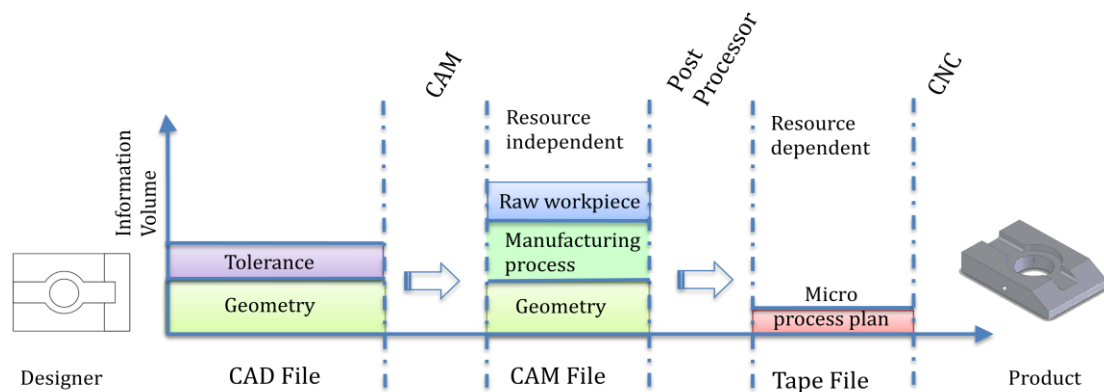


Figure 3.1 - Information lost during the process lifecycle

3.4. Part programming in relation to CAM/CNC integration

To realise system integration in CNC manufacturing, leading CNC companies have made efforts to provide practical commercial software solutions. This section will investigate the current CNC programming methods employed by the manufacturing industry to summarise the characteristics and existing problems to help identify the research gaps for this research.

3.4.1. Part programming methods: a short history and background

Generally, there are two different ways of NC part programming, namely manual and computer-assisted. The term “manual” applies to all situations in which the program is written directly in machine code by operators, even when a computer has been used. The term “computer-assisted” represents the use of computerised product design model data to generate the program utilising the computer automatically. Due to the incompatible instruction sets between different controllers, programs often have to be re-written if jobs are shifted from one machine to another, which is tedious, error-prone and costly (Groover and Zimmers 1984). Alternatively, the manufacturing enterprise loses the flexibility in the area of shopfloor scheduling (Pusztai and Sava 1983). For some complex four and five axis jobs that require accurate and consistent calculations, it is impractical to produce part programs manually. Hence, manual programming was primarily used at the early stage of the development of NC technology for small programs. With the enhancements of the computer technology, computer-assisted NC programming systems have emerged since the 1980’s to meet the developing requirements of CNC technology. The benefits of computer-assisted programming are significant in terms of programming time, accuracy and flexibility.

Computer-assisted part programming can be implemented in two different ways (McMahon and Browne 1993). The first is to use a computer-assisted programming language, and the second is to use the CAD/CAM solution. CAD/CAM has dramatically changed the way that part programs are prepared. The interactive graphics-based NC programming enables the user to easily define the part geometry, generate the tool path and obtain visual feedback immediately.

The development of computer-assisted programming languages started as early as the introduction of manufacturing systems. In the 1990’s Groover (1984) claimed there were more than 100 computer assisted languages. However, users had to prepare the representation of part geometry manually. Firstly, the part geometry had to be programmed using the computer-assisted programming language. Fortunately, using the mathematical definition of the language to program the geometry is not machine-dependent. It was the same for all different CNC machines, which certainly eases the programmer’s task. Secondly, a computer software tool is used to perform the mathematical calculations and compile a “cutter location” list or file. The processed data is then forwarded to the “postprocessor” to get the machine tool specific NC codes. There

were many computer-assisted programming languages, of which APT and COMPACT II were popular ones. The languages used “English-like” words to describe the part, tool, type of operation and the required auxiliary operations for the machining process (Pusztai and Sava 1983).

With CNC machines extensive use in manufacturing, productivity has become an important factor faced by manufacturers, and computer-aided system began to be widely adopted. In 1970s, CAD and CAM systems became commercially available. The earlier systems from the 1960s like SKETCHPAD matured into more advanced systems (Grabowski 2002). Product engineers use CAD systems to establish the part geometry, dimensions and tolerances. The design data can then be transferred into the CAM system, together with the process plan entered by the engineers, to generate the codes to machine the part. In implementation, a CAD/CAM system can consist of separate or integrated CAD and CAM software packages. This process simplifies the generation of machine codes for production of metal parts and reduced the required time and resources tremendously. For more than 30 years, CAD/CAM has proven to be a practical way for generating NC programs automatically and effectively. However, a post processor is needed to generate machine specific codes.

Postprocessors rely on an internal data model that translates cutter location data into machine controller-specific format. Due to the large number of dialects of NC standards, there has been an explosion in the number of these postprocessors. GibbsCAM (GibbsCAM 2012) claimed that there were more than 10 000 postprocessors in their library. The CAM provider has to prepare lots of postprocessors in advance. In order to limit number of required postprocessors, these CAM systems typically have a universal postprocessor. A universal postprocessor allows the end user of CAM systems to customize a postprocessor; however this is only feasible for those experienced specialists (Liu *et al.* 2007).

3.4.2. The ISO 6983 solution

The electronic files used to control CNC machines are often in a format, informally called G-codes, after Gerber Scientific Instruments, a manufacturer of photoplotters and developers of the file format (Schroeder 1998). The X-Y two-dimensional motion of photoplotters was extended to include the third Z-axis. With the addition of special

switching instructions G-codes are capable of supporting three axis milling machines. Motion commands in the control files start with the letter G.

Besides G-codes, there are usually some commands starting with M to program some other Miscellaneous functions like coolant on/off, chip removal. Hence, this standard is also referred to as G&M codes. A file containing G&M codes would comprise many lines of text that would be interpreted as moving instructions for the servomechanisms connected to various axes of the machines (Smid 2003). The language was later expanded to cover lathes and later on, the 4th and 5th axes on milling machines. With the wide use of G&M codes, it became the de-facto standard for NC programming and then was formalised as RS-274D (USA), ISO6983 (ISO) and DIN66025 (Europe) (Liu *et al.* 2007).

The major problem is, it is very common that the part programming systems on modern CNC machines significantly differ from each other even for the machines employing the same programming language, not to mention the machines with different languages. For example, due to the limitations of ISO6983, Fanuc and Siemens controllers developed their own dialects based on G&M codes. The additional instructions accepted by these controllers have never been standardised.

Currently, the famous programming system makes include Fanuc, Siemens, Mazak and Heidenhain *etc.* Fanuc and Siemens G&M commands are both based on ISO6983. Due to the deficiency of the standard, they extended the standard to develop their own dialect of G&M codes. From the comparison of Fanuc and Siemens G&M codes (GE Fanuc Automation 1998; Siemens AG 2000), the basic commands, like rapid position and linear interpolation, are the same and compliant with ISO6983. Additionally, they also follow the standard in command format. It is easy for operators to understand different G&M codes at a tool movement level. However, it is difficult to transfer the program from one CNC machine to another of a different make. This is compounded by the use of CNC proprietary languages from companies like Mazak and Heidenhain *etc.* They all accept G&M code but with their own characteristics in the formats or additional proprietary functions.

Despite the different presentations of G&M codes, in essence they are the same. They all are the motion commands for physical machine tools, which can be equipped with a controller of any make. For an identical part, similar machine tools equipped with

different controllers perform the similar or even same motions with different part programs to machine it. Hence, it is possible to be represented by canonical machining functions (Proctor *et al.* 1997; Guo *et al.* 2011).

3.4.3. Efforts in CAM/CNC integration

Due to the low level information in G&M code, there is lots of information lost at the interface between CAM and CNC. This interface is the weakest link in the CAx chain. In order to remedy this shortfall, many CNC vendors makers proposed their own solutions by developing a vendor-specific data interface for CAM and CNC.

The FAPT, abbreviation of Fanuc Automatic Program Tool, which resides in the Fanuc control, is a powerful machine programming language (Curran and Stenerson 2001). The FAPT machine control option enables the operator to program on the shopfloor according to the product design information. Using this system, the operator should use the keyboard to input the part information, the machining operations and their parameters and strategies and the tool data. Then FAPT can generate the toolpath for the part for verification, after that the final program can be generated. Generally, the FAPT is a type of vendor specific embedded CAM system on Fanuc CNC controllers. It enables the rich information flow going into the CNC. However, its final output is still the G&M codes. It can only work with Fanuc controllers. Consequently, the FAPT just provides another configuration of a CAM system and a CNC controller.

Siemens provides three programming methods: G-Code programs of CAD/CAM systems, G-Code programs manually on the machine, and ShopMill/ShopTurn shopfloor programming systems (Siemens 2009). ShopMill/ShopTurn makes programming much easier with their interactive user interface. Again similar to the FAPT, ShopMill/ShopTurn is a CAM system at the machine but only specific to Siemens controllers. Product information is input by defining the proper operations for every feature by the operator. By hiding the G&M codes from the user, it enables feature level programming and modification. Its advantage is that it simplifies the process planning process and can be G-code free ostensibly. However, likewise, it is another shopfloor level CAM system. From the product design (CAD model) to CNC, the information cannot be transferred automatically. It needs human intervention to translate the design information into manufacturing data.

Mazak, also known as Yamazaki Mazak, is another famous make of CNC controller. It employs a pre-defined operation set to compose a NC program using its own language entitled Mazatrol. For common operations, there are predefined cycles related to features such as pockets, holes, slots etc. For example, if a facing operation is needed, the operator fills in the necessary parameters (including feature geometry information, the tool information and the machine settings) according to the definition format, then a facing operation unit is added to the program. After inputting all the operation needed, an intact NC program is generated. If some special operation is needed and there is no corresponding operation defined in the set, the manual-programming mode will be used. In this mode, the programmer has to program the exact movements of all correlative axes. To specify the tool path, G&M commands are employed. When interpreted on the machine, the controller reads in the operation reference number and its parameters and decides how to move the axis to machine the feature defined. In a Mazak system, there is also a tool path control unit to edit the generated path for every operation. If it encounters with a manual unit, then faithfully follows the designated path. With the successful integration of CAMWARE (Mazak 2010) and Mazak CNC machine, a high-level feature-based integration of CAx chain is realised. It is excellent solution if you are a faithful Mazak user as it is not applicable with other CAD/CAM systems and CNC machines.

Heidenhain uses G&M codes but also has its own Heidenhain CNC language. However, Heidenhain TNC CNC series are distinct from Fanuc and Siemens for their unique programming language, which does not conform to ISO 6983. There is no G command in Heidenhain program. Instead, it uses abbreviations for every command. For example, L means Linear movement, while C represents Circular path. In Table 3.2 (Heidenhain 2009), six basic path programming commands of Heidenhain controller are shown. From the Table 3.2, it is obvious that the Heidenhain language is totally different from G&M codes. There are two kinds of movements: machining and approach & retract. All the movements are straight lines or circles or the combination of lines, circles and approach/retract connections.

Table 3.2 - Examples of Heidenhain Programming Language Commands

Abbreviation	Meaning
APPR	Approach
DEP	Departure
L	Line
C	Circle
T	Tangential (smooth connection)
N	Normal (perpendicular)

Heidenhain is a feature based programming language that uses its own comprehensive set of canned cycles. It pre-defines the corresponding cycles for most common features (e.g. pocket, drilled holes, etc). Combined with the basic movement commands as the connection, these cycles can make up a different format NC program. Hence, this programming language carries more information than G&M codes. The CNC knows what feature is going to be machined on the machine. Additionally, the introduction of smarT.NC programming system (Heidenhain 2009), again a vendor specific CAM-CNC system, takes the programming task away from CAM office to the shopfloor creating a certain level of flexibility in CNC machining.

Though there are many differences between these programming systems, most of them program tool movements. The machine tool is only an obedient executor. It knows “how to do” but not “what to do” (Xu and Newman 2006). Although the CAM/CNC vendors have provided proprietary solutions to pass part geometry information to CNC controllers, it is only in the operation/feature format. To some extent, it is like the canned cycle in G&M codes. Moreover, it is difficult to program automatically by computer and not exchangeable between different vendors. In Table 3.3, the four different programming systems are compared in terms of programming language, workpiece definition, part geometry, operation and toolpath.

Table 3.3 - Contrasts between different systems

	Programming language	Workpiece definition	Operation/ Feature	Toolpath	Exchangeability with others
Fanuc	FAPT + ISO 6983 (Dialect)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Siemens	Siemens	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Mazak	Mazatrol	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Heidenhain	Heidenhain	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

3.5. State-of-the-art of in CAD/CAM/CNC Interoperability

Interoperability is the ability of two or more systems or components to exchange information and to use the information that has been exchanged (IEEE 1990). Generally, interoperability is a higher level of integration. It describes the capability to exchange information between different systems while maintaining its original set of semantics (Nassehi 2007). Ray and Jones (2006) defined interoperability as the capability to realise the error-free transmission and translation of information between different computer systems without manual intervention. In manufacturing, interoperability is a long pursued goal that has not been achieved in practice (Martin 2005). The dynamic response to the changing market is still limited by the paucity of interoperability (National Coalition for Advanced manufacturing 2001; Brunnermeier and Martin 2002). According to the study of NIST (National Institute of Standards and Technology), the U.S. automotive sector alone expends one billion dollars per year to resolve interoperability problems. (Brunnermeier and Martin 1999).

In this section an overview of the literature and commercial solutions on interoperability has been provided. The literature is categorised based on the enabling technology. Critique of the implementation methods is summarised.

3.5.1. STEP-NC enabled manufacturing interoperability

3.5.1.1. An introduction to STEP-NC

STEP-NC, formally referred as ISO 14649 (ISO 14649-1 2002) and ISO 10303 AP 238, has been developed since late 1990s aiming to remedy the shortcomings of ISO 6983 to support the comprehension information exchange between CAM and CNC. As the new data interface for CNC, one of the expected benefits from STEP-NC is bringing the component geometry into the controller. However, STEP-NC goes much further and is defined as a data model for next generation CNC controllers, with the aim to overcome the lack of process planning information in ISO 6983 files (Rosso Jr 2005). STEP-NC contains comprehensive information including, process sequence, raw material definition, manufacturing feature, machining operation, machining strategy, cutting tool geometry etc. An overall structure of ISO 14649 is illustrated in Figure 3.2.

As the basis of STEP-NC, ISO 10303 or STEP is the standard for product information exchange throughout the product lifecycle. STEP aims to provide a standardised, neutral data format for product information exchange and sharing between different CAX systems. One of the most important aspects of STEP is its extensibility, which is achieved through the use of EXPRESS (ISO 10303-11 1994) as the data modelling language (Loffredo 2000). Another characteristic of STEP differentiating it from other standards is the independence of data model defined in the standard and the implementation methods (Fowler 1995).

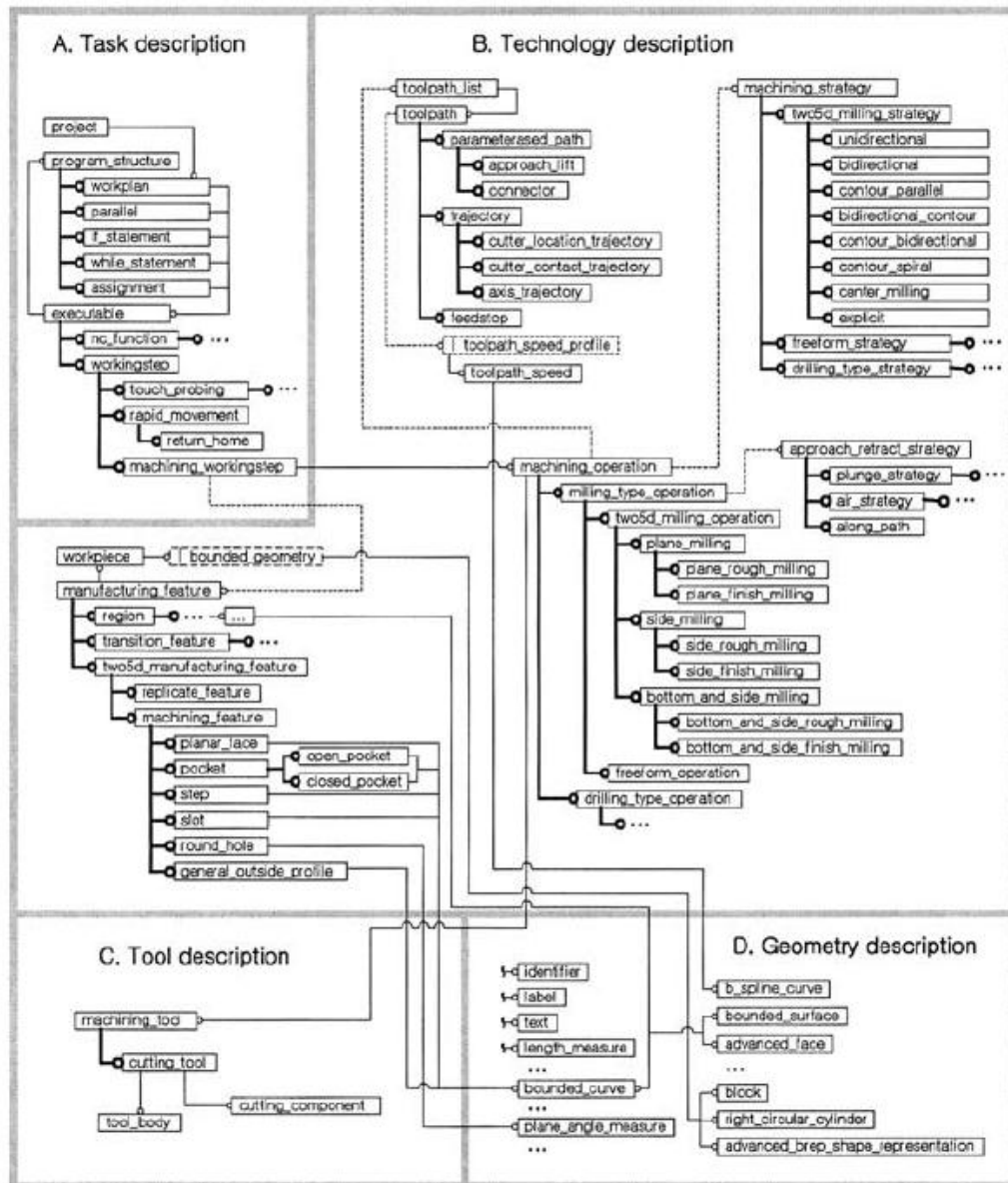


Figure 3.2 - The overall structure of ISO 14649 (Suh *et al.* 2002a)

STEP-NC, both ISO 14649 and ISO 10303-238, aims to provide a standardised numerical control data for CNC controllers. ISO 14649 uses the EXPRESS language and corresponding implementation methods from STEP to define a set of NC domain requirements models in terminology familiar to NC user. ISO 10303-238 (hereafter AP 238), Application Interpreted Model (AIM) for computer numeric controllers, is an Application Protocol (AP) in STEP family of standards. In AP 238, the EXPRESS models in ISO 14649 are used as the domain requirements model (called an Application Reference Model, or ARM) with a few modifications then mapped to the STEP integrated resource to obtain the implementation model (Application Interpreted Model, or AIM). “The primary difference between the two standards is the degree to which they use the STEP

representation methods and technical architecture. (Feeney *et al.* 2003)” AP 238 is more compatible with STEP standard. A major issue in the implementation of STEP-NC is, which is better, ISO 14649 or AP 238. A discussion on the advantages and disadvantages of the two standards can be found in Feeney *et al.* (2003) and Wolf (2003). According to Kramer *et al.* (2006) and Rosso (2005), AP 238 fits better together with other parts of STEP. However, the ISO 14649 solution provides better support and the basic level of information requirements for CNC controllers (Feeney *et al.* 2003). The data format of ISO 14649 is not so complex, and using ISO 14649 can avoid the performance sacrifice of complex AP 238 data (Kramer *et al.* 2006).

ISO 14649 is comprised of various parts while each deals with a specific CNC technology. The prominent parts of STEP-NC are:

a) ISO 14649 part 1: Overview and fundamentals

As the basis of the ISO 14649, this part contains the overview and fundamental principles for the standard. It describes the context of the standard and introduces the relationship between ISO 14649 and ISO 10303. The whole architecture and development schedule of the standard are also included in this part (ISO 14649-1 2002).

b) ISO 14649 part 10: General process data

This part of ISO 14649 (ISO 14649-10 2002) contains the representation model for generic process data which is generally needed for NC programming within different machining technologies. The fundamental entities like manufacturing feature, the control structure of the program including project, workplan and workingstep *et al.* are defined in EXPRESS schema.

c) ISO 14649 part 11, part 12: process data

These two parts (ISO 14649-11 2002; ISO 14649-12 2004) specify the machine independent, technology-specific data elements needed as process data for milling and turning respectively. They mainly focus on the entities representing the manufacturing process as generic process plan. Together with the basic entities, these parts describe the programming interface for a computerised numerical controller.

d) ISO 14649 part 111 and part 121: Tools for milling and turning

These parts of ISO 14649 (ISO 14649-111 2002; ISO 14649-121 2002) contain entities to describe manufacturing tools for milling and turning respectively. They work together with ISO 14649-11 and ISO 14649-12.

3.5.1.2. STEP-NC research review

(i) Intelligent manufacturing and the CNC controller

The introduction of STEP-NC is not merely another interface between CAM and CNC, but also an immense challenge and opportunity for modern manufacturing and numerical control technology (Xu and He 2002). By replacing G&M codes, STEP-NC is thought to bring great changes into the whole manufacturing industry. It will provide the next generation of manufacturing system with openness, integrity, adaptability, interoperability to realise intelligent, distributed control and manufacturing (Xu and Newman 2006). A significant body of research has been carried out on the next generation CNC controllers based on STEP-NC with the goal of intelligent manufacturing and is outlined below. In this section, a review of these research has been conducted to identify the disadvantages of the new CNC controller development in relation to shopfloor interoperability.

Basically there two types implementation methods of STEP-NC: indirect interpretation of STEP-NC on conventional CNC controllers and adaptive STEP-NC controllers (Zhang *et al.* 2006; Rauch *et al.* 2012). The indirect implementation method is interpreting STEP-NC data and post-processing into native part programmes or control signals. It does not need to develop new controllers; instead an interpreting module is added to current CNC controllers. The interpretation module can be embedded in or interfaced with CNC controllers. The adaptive CNC controllers can interpret the STEP-NC data directly to drive the machine tools to perform adaptively according to the feature geometry and process information. There is a need to develop new STEP-NC controllers.

The indirect method is a practical and initial solution to benefit the innovations of STEP-NC. Xu (2006a) presented a G-code free machining scenario by implementing STEP-NC on a legacy CNC system of a lathe. In this research, STEP Tools STIX (STEP Tools Inc 2004) has been used to read and process STEP-NC information. STIX, STEP Index Library, is a C++ library providing useful functions to extract manufacturing information from STEP AP 238 format files. Then, a machine specific low-level NC program for the CNC

system is generated by the interface STEPcNC based on the manufacturing information contained in STEP-NC. This is a typical example of the first type of implementation of STEP-NC.

Another example of indirect implementation of STEP-NC is conducted by Rauch et al. (2012). With the current manufacturing resources STEP-NC can be used to explore the opportunities of intelligent high level programming and adaptive control according to the machining conditions. A STEP-NC platform for advanced and intelligent manufacturing (SPAIM) is proposed. It has several modules to interpret STEP-NC data, carry out tool path planning, parameter optimisation, simulation, inspection etc. Current CNC controllers can be interfaced to this platform to become STEP-NC compliant. The platform has been implemented and validated on two commercial CNC machines with a Siemens Sinumerik840D controller and a Heidenhain CNC controller.

For the direct implementation of STEP-NC, a new generation of CNC controllers need to be designed. Based on the analysis of the STEP-NC information content and the role of CNC on the shopfloor in an intelligent manufacturing system, Suh and Cheon (2002) proposed a conceptual framework of next generation CNC system. Then the framework was extended to include an implementation method for a milling machine called ASNC, Autonomous STEP-compliant CNC(Suh *et al.* 2002a). It takes ISO 14649 as input and carries out manufacturing tasks in an intelligent and autonomous manner. With the input of STEP-NC, there is an interpreter in this framework to read in the manufacturing information. Other function modules use this information to plan details (e.g. toolpath) for the job. Finally, the control information is sent to the control modules to machine the part. To realise the autonomous control, there are an operation monitor and an online inspector to feedback the status to the controller. This implementation method was then realised on a prototype milling machine (Suh *et al.* 2002b). In order to support and integrate with the new controller, a shopfloor programming system, PosSFP, was proposed by them (Suh *et al.* 2003). It can recognise features from the CAD file, generate process plans and then generate the complete STEP-NC file. Together with the controller, it enables a STEP-compliant based CAD/CAM/CNC chain solution.

Lee and Bang (2003) developed a STEP-NC milling controller using a PC and a motion control board. The input of this milling machine is the ISO 14649 file in XML format. The main modules of the system are the STEP-NC interpreter and the toolpath generator.

With the actual machine, an example part was machined using the program in the appendix of ISO 14649-11 (2002). Through this process, STEP-NC enabled CNC machining was realised. It proved the possibility to use STEP-NC to integrate the CAD/CAM and CNC. The adoption of XML coding of STEP-NC data facilitates the system with easy adaptation with other web-based systems or services.

Calabrese and Celetanno (2007) gave a practical and economic solution to realise a STEP-NC controller by retrofitting an existing CNC system. The detailed design and realization of the system including the hardware and software were presented. This architecture is actually a CAM system embedded in a conventional CNC controller, and meanwhile, it is a practical way to implement STEP-NC at the initial stage, and is claimed to smooth the stage from G&M codes to STEP-NC.

Zhang et al. (2006) provided a futuristic version of an autonomous STEP-NC controller. Based on the discussion of the strategies to implement STEP-compliant CNC system, three potential implementation options were given according to their different function level and the degree of adaptability, namely: interfaced, CAM-embedded and intelligent STEP-NC controller. Through analysis, the research concluded that the architecture of ADACOR agent (Leitão and Restivo 2006) well fits the function requirements of intelligent STEP-NC controller. A framework for autonomous STEP-NC controller based on ADACOR architecture was proposed. A complete implementation of this design could enable interoperability within the domain of CNC machines, as a controller can make manufacturing decisions autonomously based on its configuration.

Li et al. (2012) proposed a novel framework for STEP-Compliant and knowledge-base supported intelligent CNC controller. It adopted ontology technology aiming to solve the problem of knowledge sharing, integration, interoperability and reuse. The CNC controller framework utilises a Three-Layer Ontology-Based Knowledge Model. In the design of the knowledge base, a semantic markup language Web Ontology Language (OWL) is used to facilitate publishing and sharing ontologies on the World Wide Web. A step-compliant CNC controller is presented and the work flow is introduced.

All the research in this section focuses on the development of new CNC controllers adopting STEP-NC data. The disadvantage is there is a need to retrofit the current CNC machines with new controllers or have new CNC machines with STEP-NC controllers. Both the controller vendors and the users have to make efforts to adopt the new

controllers to take advantage of STEP-NC, which is unlikely to be realised in the short term.

(ii) Information flow and system integration

STEP-NC provides an information exchange framework that enables the interoperability between various systems. Considerable research has been conducted on the system integration and interoperability issue based on STEP-NC. In this section, a brief review of the research have been provided to identify the issues with this method.

Based on the illustration of interoperable manufacturing scenario by Boeing and NIST, Hardwick and Loffredo (Hardwick and Loffredo 2006) published a paper documenting the test consisting of four CAMs and two different five-axis CNC machines. The AP 238 file was converted into machine specific tool path codes to control the machine. This demonstration showed that it is not so difficult for CAM and CNC vendors to implement STEP-NC AP 238.

Ali et al. (2005) proposed a STEP compliant inspection framework for a component, aiming to provide a capability to establish standardised measuring and inspection across the total CAX chain to close the loop and feedback of inspection results to component model design. In order to achieve this integration STEP-NC (ISO 14649) and AP219 have been used to provide the basis of a STEP-compliant inspection framework.

Brecher et al. (2006) introduced the integration of measuring technology into the STEP-NC based process chain, to be able to preserve the results of the manufacturing process in a set of data and feed them back to the planning process. A prototype demonstration case for the closed-loop process chain was presented, which includes generation and execution of a STEP-NC program and feedback of measured results to the CAM system.

Wosnik et al.(2006) gave a generic approach to pre-process and feedback process data from servo drives to CNCs and CAPP systems. The design of application-dependent algorithms can process and exchange drive signals for both, online and offline optimization of machining processes.

The eXtensible Markup Language (XML) is emerging as the primary translation media for transferring information across the internet (Cang *et al.* 2006). Some manufacturing

experts are devoted to expressing and transferring the manufacturing information in the XML format. Due to the ever-increasing competition market environment, globally distributed manufacturing or e-manufacturing is a major trend. A reliable and robust information exchange through internet is necessary to realise the interoperability between remote systems. Cang et al. (2006) suggested XML as the suitable file format for carrying STEP-NC information across the enterprise-wide information network in the e-manufacturing scenario. Lee and Bang (2003) developed a STEP-NC milling controller with an XML interpreter to deal with STEP-NC information in XML format. Borsellino et al (2004) illustrated remote operation of an AP 238 file in XML format for internet based machining. From the results of the research utilising XML file, it is clear that XML is a suitable data format in expressing and transferring manufacturing data to facilitate the realisation of interoperability.

Nassehi et al.(2006a) explored the application of mobile agents to realise the interoperability in a global manufacturing environment. A novel manufacturing chain based on the STEP-NC standard was proposed and the application of mobile agents based multi-agent system was studied with a prototype provided.

Newman and Nassehi (2007) proposed a Universal Manufacturing Platform using STEP-NC as the enabler to realise a standardised information communication platform for various applications. It acts like an operating system of a normal computer where the software applications can access the hardware and software in an abstract manner. They claimed not to need the information about the resources it is going to use and the operation system will handle the connection between them. UMP can be considered as a manufacturing knowledge and information support system that aims to accommodate not only the CAx systems but also other business application systems employed in the enterprise. There are three components in UMP: the intercommunication bus, the manufacturing data warehouse and the manufacturing knowledgebase. To implement a UMP, a systematic roadmap is proposed by Mokhtar and Houshmand (2010) using the methodology of Axiomatic Design.

Campos and Miguez (2011) examined the possibility to take advantage of STEP-NC to enable the a collaborative manufacturing scenario with programmed traceability and monitoring services. An extension to STEP-NC with new nc_functions has been proposed for programming process monitoring and traceability. A prototype scenario based on this

extended model has been implemented with a description of how the standards (ISA-95 and MTConnect) may support and complement STEP-NC to establish a collaborative STEP-based CAD/CAM/CNC supply chain to program and automate machining process data monitoring and traceability activities.

The research works in this section provide a future version of system integration and interoperability using STEP-NC. However, in the current stage since there is no commercial STEP-NC information systems available, it is still too early to take advantage of STEP-NC to facilitate the integration and interoperability between different systems.

(iii) Computer aided process planning

In CNC manufacturing, process planning is an information intensive and intelligence requisite activity. The new programming interface for CNC (STEP-NC) proposes new challenges for process planning technology and provides an interesting area for the application of artificial intelligence, knowledge-based system and recently agent technologies.

Newman et al.(2003) gave several possible generic frameworks for how CAD/CAM systems may evolve using ISO 14649, and presented an Agent-Based Computer Aided Manufacture (AB-CAM) system. This system combines STEP-compliant, feature-based design with agent-based computer aided process planning. Allen et al.(2005) gave the detail implementation of multi-agent process planner of the AB-CAM system and brought Manufacturing Feature Agent (MFA) into process planning.

Suh et al.(2003) proposed an architecture for Shop Floor Programming (SFP) system using non-linear process planning technology. Then they extended this area in detail by development of an optional solution algorithm for process planning of complex parts (Chung and Suh 2008). This nonlinear process planning is composed of three parts: Neutral Process Sequence Graph (NPSG), Hardware-dependent Process Sequence Graph (HPSG) and Executable Process Sequence Graph (EPSG). The NPSG is responsible to extract neutral process information from the text STEP-NC program in AND-OR graph. While in the HPSG, hardware information is added into the process graph. The optimal solution algorithm is used to determine the best process (EPSG) from the possible choices of HPSG, while the nonlinear process is linearised to be executable on the specific CNC controller. Then, Shin et al.(2009) employed the shopfloor programming system to

support the multi-channel complex machine tools, named e-CAM system. The shopfloor programming system is a typical CAM solution in the new STEP-compliant manufacturing environment.

Xie and Xu (2006) reported a STEP-compliant computer aided process planning prototype system for sheet metal products. The system has an interface to input CAD product data. It integrates software modules for nesting optimization, path optimization and planning, simulation, and machining parameters set-up and CNC machining. This framework divides information into four layers, i.e. a knowledge layer, a product layer, a feature layer and a parametric layer. It is an important step towards the development of a fully integrated CAD, CAPP, CAM and CNC manufacturing chain.

Nassehi et al. (2006b) outlined a multi-agent process planning prototype system for prismatic components named Multi-Agent System for Computer Aided Process Planning (MASCAPP). It investigated the application of distributed artificial intelligence methods, namely collaborative multi-agent systems in designing an object-oriented process planning system for prismatic components in a STEP-NC compliant environment. They concluded that intelligent agents and distributed artificial intelligence show a lot potential in process planning systems and agent systems fit well with STEP-NC, as it can be served as an excellent knowledge database for its comprehensive product information.

Amaitik and Kiliç (2007) presented a hybrid intelligent process planning system (ST-FeatCAPP) using STEP features for prismatic parts to integrate CAPP and CAD systems. Using a feature modeller (STEP-FM), the design data can be mapped into a STEP AP224 XML data file, and the corresponding machining operations can be identified to generate the process plan and corresponding STEP-NC in XML format. During this process, the complex feature recognition task can be avoided. The overall structure of this system can be seen in Figure 3.3. In implementation, a hybrid approach of most recent techniques (neural networks, fuzzy logic and rule-based) of artificial intelligence is used as the inference engine of the developed system.

Zhao *et al.* (2009) painted a typical STEP-compliant manufacturing environment, which effectively integrates two sub-systems: process planning and manufacturing. The process planning system is used to recognise features from generic design data and store the feature information in a STEP compliant format (AP224). Standardised feature based

information are then passed into macro and micro process planning activities to generate generic and native STEP-NC programmes respectively. “In such a system, the need of data conversion is eliminated” (Xu *et al.* 2011).

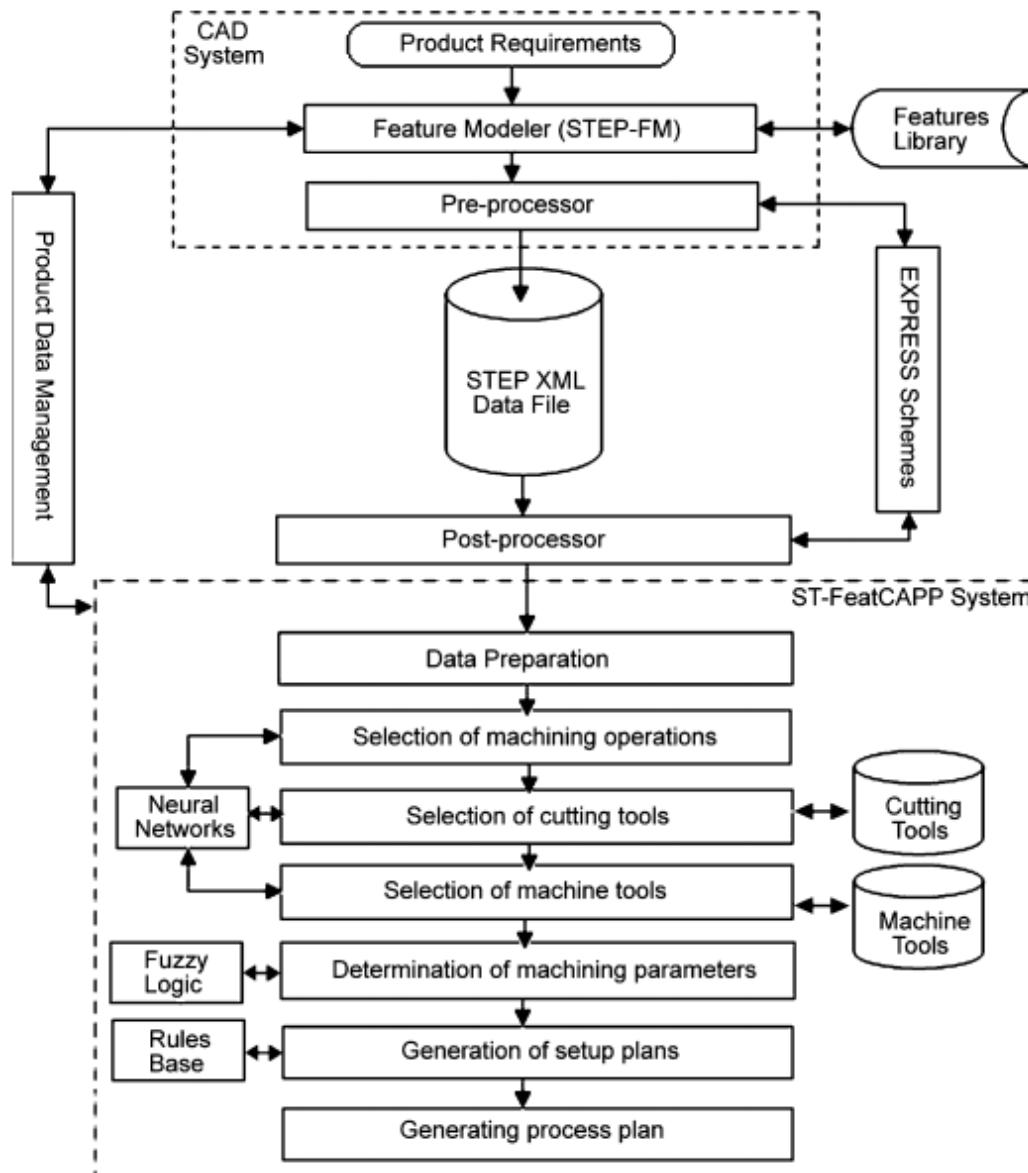


Figure 3.3 - ST-FeatCAPP architecture (Amaitik and Kiliç 2007)

Nassehi et al.(2007) introduced a novel software platform entitled Integrated Platform for Process Planning and Control (IP³AC) to support STEP-NC compliant process planning. A prototype of process planning system based on this platform was proposed. IP³AC provides a fully object-oriented encapsulation of ISO 14649 information. Hence, it is easy to expand the upper process planning system to handle more complex product

without any change to the IP³AC. This platform was the basis for the UMP research mentioned earlier in this section.

Zhang et al (2011a) proposed a STEP-NC process planning system for turning operations. This system is supposed to act as a module of intelligent CNC controller. It enables the operator to make the last minute changes to the parameters at the shopfloor level. It receives STEP-NC programs and performs some modifications and optimisations before generating the appropriate tool path. Surface roughness is chosen as the optimization object to adjust the detailed parameters.

Instead of developing process planning systems, Shin et al. (2007) proposed a G2STEP system to capture the process plan in existing part programmes for turning operations. It recognised the features and operations and reconstruct the process plan in STEP-NC format. It provided a practical way to accumulate shopfloor knowhow from real machining practices and facilitate the repaid development of process plan.

The above papers show that utilising STEP-NC in process planning systems can eliminate the problem, such as feature recognition, of the integration between CAD and CAPP/CAM systems. However, it still needs the CAD vendors to make their efforts to support STEP-NC and new CNC controllers need to be developed to accept STEP-NC data. Nevertheless, Shin et al. (2007) does provide an innovative way to take advantage of STEP-NC in the current stage and it is practical as well to capture the knowledge in the part programmes. However, this approach only focuses on turning and does not consider milling and other operations as identified in the author's research scope in Section 2.3.

(iv) Investigation on the performance and feasibility of STEP-NC as interoperability enabler

Since the STEP-NC emerged, several review papers documenting the advances and guiding the development of STEP-NC have been published. They investigated the performance and feasibility of STEP-NC as the enabler of interoperability. In this section, the performance of STEP-NC has been discussed.

Xu and He (2002) ascertained that STEP-NC would replace G&M codes and reshape manufacturing. Two years later, Xu and He (2004) reviewed major global endeavours in STEP-NC related research and identified the benefits, opportunities and challenges concerning STEP-NC development. Based on the earlier reviews, Xu et al. (2005)

presented a comprehensive review of STEP-NC developments, including the international research projects and the research carried out by major research groups in the world. This paper also gave a futuristic view of CAD, CAPP, CAM and CNC integration based on STEP-NC.

Regarding the implementation stages of STEP-NC, Suh and Lee (2004) presented a STEP-manufacturing roadmap consisting of three steps as the specific approach methodology to formalize a STEP-manufacturing environment: 1) participation with many manufacturing related companies, 2) inducement towards an information oriented and international environment, and finally 3) consideration of compact and economical research and development.

Xu and Newman (2006) examined different STEP-NC implementation methodologies to make CNC machine tools more open, interoperable and intelligent. The major impediments of current CNC technology were discussed and a STEP-compliant collaborative manufacturing model was proposed.

Newman et al. (2008) reviewed the evolution of CNC manufacturing and interoperable manufacturing, gave a strategic view of how interoperability can be implemented across CAx chain. This paper provided the strategic advantage of manufacturing interoperability for enterprise and identified the major challenges of the implementation. It implied that the current software/hardware vendors were the major barrier due to business benefits. Another reason is the inherent complexity of STEP compliant CNC controller. Hence, a global interoperable STEP-NC system is still a long way away.

To exploit STEP-NC to enhance the current CNC manufacturing interoperability, a hybrid system (Rauch *et al.* 2009) of two manufacturing platforms: STEP-NC Platform for Advanced and Intelligent Manufacturing (SPAIM) and Intelligent and Interoperable Manufacturing Platform (IIMP). SPAIM is able to control current industrial machine tools directly from STEP-NC files and IIMP realizes data portability between heterogeneous proprietary formats, process interoperability. The combination of these two platforms is aiming to realise an improved supervision and integration of the machining systems with STEP-NC standard.

From the review papers in this section, STEP-NC is identified as a good solution to the current interoperability problem of the CNC manufacturing chain. The implementation

still needs a huge amount of work to develop new CAx systems and to make modifications to the existing resources. Since there is no commercial benefits for the controller and software vendors to make efforts to adopt STEP-NC, the realisation of a STEP-NC interoperable system is still unachievable in the near future.

3.5.2. Other technologies based interoperability

Various approaches for implementation of interoperability have been explored by researchers. The straightforward way is to make use of the low-level language currently utilised in the manufacturing industry. Researchers have presented several automatically translation mechanisms between different languages to realise interoperability without modifying the current standards.

In order to reuse NC programs on different CNC systems, Liu et al.(2007) and Guo et al. (2011) proposed a NC Program Processing system called NCPP that decodes the NC part program (G&M codes file) into a neutral representation named canonical machining functions such as motion command and Programmable Logic Controller (PLC) commands. These commands can be sent to PLC of CNCs and the motion controller processor separately to control the cutting tool movements and auxiliary devices.

Schroeder and Hoffmann (2006) also presented algorithms to realise the conversion of RS-274D compliant NC programs from one type to another. RS-274D is a USA equivalent standard for NC part programming. The results showed that using these algorithms 90% of complex programs could be converted automatically while the remaining 10% need human intervention.

Spence et al. (2000) proposed to use Cutter Location Data (CL-DATA) to develop an integrated solid modeller for a comprehensive physical machining process simulation. A prototype system of the solid modeller has been developed and tested. The input of the modeller includes the CL-DATA and the raw workpiece information. Based on that a solid model of the product can be derived for engineering analysis such as tool and part deflection. Although it is designed for process simulation, it is can be possibly used to realise the communication from the CNC machine to CAD systems. However, it still needs to further efforts as it accepts real part programmes from the machines instead of the CL-DATA. The solid model will still need to be machined on another machine. Going from a solid model to a machining programme is non-trivial.

The incomplete interoperability achieved by such approaches (Liu *et al.* 2007; Schroeder and Hoffmann 2006) is not a long-term solution. They are still based on the low-level information-containing standard. They only focus on the CNC link of the chain and cannot solve the information isolation problem in the entire CAx chain. Spence *et al.* (2000) only focused on solid model without considering process knowledge. For more advanced interoperability, a new data model standard of the CAx systems is needed to provide the wide and solid fundamental bases for constructive solutions. Martin (2005) explored the opportunities of international standards as the enabler of the interoperability in manufacturing enterprises and supply chain, as reviewed in the previous section. Instead of focusing on the information model, other researchers have paid more attention on the information sharing mechanism towards global distributed interoperable manufacturing. Recent advancements in other subjects such as Computer Science, System Engineering have consequently been introduced into manufacturing. These technologies include agent-based technologies, information technology, semantic interoperability, web-based communication and enterprise integration etc.

Peng *et al.* (1999) explored an agent-based approach towards intelligent manufacturing integration. A multi-agent system supporting the integration of manufacturing planning and execution was proposed. The Knowledge Query Manipulation Language (KQML) was used as the communication language and protocol between agents. KQML is both a message format and a message protocol to support run-time knowledge sharing between software agents (Finin *et al.* 1994). It considers that each message not only contains the content but also the attitude or “intention” the sender has for that content. It helps to ensure the sender and the receiver have the same understanding of the message. This characteristic of KQML is thought to contribute the comprehensive information interoperability.

Ong (2002) proposed web-based Virtual Manufacturing (VM) system for CNC milling operation using Virtual Reality Modelling Language (VRML) and Java. Its open modularised design of this system would enable users to do special customisation to meet different needs.

With the goal of providing a seamless interoperability throughout the entire enterprise, Gao *et al.* (2003) presented a software solution for the exchange of product data between different departments. The solution provided analysable product information at the

conceptual stage of product design and manufacturing evaluation utilising a Product Data Management (PDM) system and STEP compliant product data model.

Shen (2005) reported a framework called iShopFloor for an intelligent internet-enabled shopfloor control system encompassing both information architecture and integration methodologies. The adoption of XML (eXtensible Markup Language) facilitates the integration this agent-based system with other shopfloor resources for its simplification and standardization of message services in Internet-based intelligent shopfloors.

Agent-based technology is an important branch of artificial intelligence and has been widely introduced and employed in manufacturing applications for its autonomy, flexibility, re-configurability and scalability (Zhang *et al.* 2011b; Wang *et al.* 2009; Zhang *et al.* 2010). Nassehi (2006a; 2006b), as described earlier, proposed to use agent-based technology to realise the interoperability between manufacturing systems. Zhang *et al.* (2011b) put forward an agent-based smart objects management system for ubiquitous manufacturing. The smart objects are equipped with heterogeneous RFID devices, which enable real-time traceability, visibility and interoperability in improving the performance of shop-floor planning, execution and control.

3.5.3. Commercial solutions on manufacturing interoperability

In order to solve the shopfloor interoperability problem, some commercial companies are exploring practical solutions and have commercial products available. This section presents a review on these commercial solutions.

CGTech (CGTech 2012) is a leading company specialising in numerical control (NC/CNC) simulation, verification, optimisation, and analysis software technology for manufacturing. Its main product VERICUT is widely used by many manufacturing enterprises. It claims to accomplish many different functions including verification, machine simulation, optimisation based on the NC part programmes. Regarding the interest of this research, VERICUT can be a reverse post processor. "By translating NC programs to APT or other NC data formats, it saves valuable machine tool and programmer time and makes it possible to recycle obsolete or incompatible NC programs." (CGTech 2012)

Predator Software (2012) is an engineering software company focusing on the development of applications and solutions that provide manufacturers with answers to

the real world shopfloor problems. One software product of the company is called Predator Virtual CNC, which simulates and verifies operation of CNC machines offline. In addition to that, it can translate G-code from CNC machines to universal toolpath, which is a proprietary NC data format. Using another product named Predator OutPost to convert the universal toolpath to G-code for other CNC machines. The data converted in this process is the syntax (toolpath in the part programme) not the semantics behind the part programme.

Actually the two solutions mentioned above to solve the interoperability problem are quite similar. They read in part programmes, store the toolpath and convert it into other programming dialects for other CNC machines. The case is quite similar to the literature (Liu *et al.* 2007; Guo *et al.* 2011; Schroeder and Hoffmann 2006) mentioned in the previous section. However, the literature and the commercial products are all assuming that the two different machines involved in exchanging part programmes have the same physical axis configuration and use cutting tools with the same diameters and cutting heights. Going beyond any of these assumptions would require the toolpath to be generated again in the CAD/CAM systems.

3.6. Critique

In this chapter, an overview of NC/CNC technology and the CAx chain interoperability has been presented. There are many innovative techniques and approaches that have been adopted across the world to enhance the integration and interoperability of different systems involved in manufacturing. Based on the analysis of the literature, two major research gaps have been identified:

3.6.1. CNC machines in CAx integration and interoperability

In state-of-the-art CNC manufacturing, the information flow is predominantly unidirectional. From the design through the CAx chain to the final part, during this process, necessary information is added at each stage to generate the detailed toolpath command to execute on the machine. The part is the ultimate object. However, the program generated by the postprocessor is the de facto final aim, where all the information including the original design and the later added information is lost. The most important final step, shopfloor machining, is isolated from the former systems. The

literature on interoperability did not pay enough attention on this most important link of manufacturing.

3.6.2. The requirement for modification of current manufacturing resources

The emergence of STEP-NC has given a promising future for a standardised solution for interoperability in manufacturing. The ideal solution is for all the systems involved in CNC manufacturing to adopt STEP-NC to realise the full interoperability. However, there is no commercial motivation for major CAD, CAM and CNC vendors to implement interoperability. The sole beneficiary is the manufacturing user. Especially for the CNC vendors, it will be a burdensome task to implement STEP-NC. The new controller should cover some other tasks that are currently carried out by CAM system. Thus there is a strong resistance with the wide implementation of STEP-NC.

Under this situation, a practical interoperable solution without the requirement for modification of current resources is required. Since there is no need for the vendors to make change to their products, the realisation of interoperability can be smoothly achieved without much resistance.

4. CAPP and knowledge reuse in CAx manufacturing chain

4.1. Introduction

This chapter reviews the literature on manufacturing knowledge reuse. As knowledge management is a broad across-domain subject, this research only focuses on the manufacturing knowledge issues. The state-of-the-art of manufacturing reuse has been presented to identify the current research gap on shopfloor knowledge capture and reuse. To answer the question of why knowledge reuse is necessary, a short review on CAPP and feature recognition is also provided to highlight the value of process knowledge existing in part programmes.

4.2. Computer Aided Process Planning (CAPP)

CAPP is a vital link in automated manufacturing. The integration issues between CAPP and other systems are still not effectively solved. Since CAPP systems are not as sophisticated as CAD/CAM systems, the development of a suitable process plan is not as easy as the development of a CAD model. In most cases, it is still developed manually by the process engineers utilising their experience, expertise and knowledge. In this section, a short review of current CAPP techniques and problems has been conducted to underline the valuable knowledge and how-how contained in a good, workable process plan.

4.2.1. Background

According to the definition of Society of Manufacturing Engineers, process planning is the systematic determination of the methods by which a product is to be manufactured economically and competitively (Alting 1989). It plays a critical role in the manufacturing process, linking design and manufacturing. With the increase in competitiveness of global marketing and complexity of discrete parts, process planning becomes more and more important for manufacturing industries, and attracts research attentions world widely.

Process planning, in terms of machining processes, involves several major activities, including interpretation of product design data, analysis of part requirements, preparation of raw material, selection of machine tools, cutter path planning, fixture, determination of datum reference surfaces, sequencing of operations, cutting tools, cutting conditions,

calculation of cost, energy, production time etc. and generation of the process sheets (Alting 1989; Cay and Chassapis 1997; Bourne *et al.* 2011).

The traditional way to solve process planning problems is to leave it to the manufacturing experts. The disadvantage of this approach is that it is time-consuming and the quality of the developed process plans may not be optimum, which depends on the process planner's experience and knowledge of the manufacturing resources. Due to the deficiency of skilled process planner, Niebel (1965) first proposed the possibility to use computer to automate the process planning task. Compared with the traditional approach, a CAPP system contributes to the systematic development of accurate and consistent process plans, helps to reduce the leading time of new products and cost of process planning, and easily interfaces with other higher level application packages such as time or cost estimation. Due to the promising benefits, CAPP has received lots of research effort and has a prolonged and prolific history of research and publications. Several reviewers (Alting 1989; Gouda and Taraman 1989; ElMaraghy 1993; Eversheim and Schneewind 1993; Xu *et al.* 2011) have reviewed these publications and hundreds of CAPP systems.

4.2.2. Scope of process planning

Process planning for machining processes received loads of attentions due to its widespread use and importance (ElMaraghy 2007). It is the typical mean of the terminology CAPP as well. In this research, by process planning below, it focuses on the issues in the domain of CNC machining.

However, process planning could be a wide cross-topic concept. The ever-increasing global competition drives manufacturing enterprises to reduce the cost of the total process from the raw material to assembly or even to the services. Process planning techniques are being used to other non-traditional domains such as assembly/disassembly, inspection, robot tasks, rapid prototyping etc. (Eversheim and Schneewind 1993; ElMaraghy 2007).

4.2.3. Classifications of CAPP

According to the implementation strategy, CAPP systems can be classified into two types: variant and generative (ElMaraghy 1993; Maropoulos 1995a). Variant approach is basically a database retrieval method based on the understanding that similar parts will

be machined in similar plans. Similar parts are grouped into a product family. For each family, there is a standard process plan developed and stored in the database. For a new part, the planning process is identifying the product family, retrieving the process plan and modifying the process plan to suit the new part. The advantage of this approach is that it needs low manpower consumption to offer a reliable and reasonable solution in real production. It is very attractive to small and medium size companies (Alting 1989). The shortcoming is that it still needs human intervention, and the preparation of the database needs substantial human efforts.

However, in a generative approach, there is little need for human intervention. Instead of retrieving existing process plans, generative approach creates the process plan by means of decision logic and process knowledge (Alting 1989). Although it has the advantage of fully automation, a generative process planning system relies on two ingredients: technical knowledge of manufacturing and computer compatible description of the part. These two elements are also the bottleneck of this approach. This is why both feature technology and knowledge-based techniques have been heavily researched in association with computer-aided process planning (Xu *et al.* 2011).

4.2.4. Current situation of CAPP

Although tremendous efforts have been made on CAPP, the benefits of CAPP in real industry are not matched with the research efforts. CAPP has not kept pace with the development of CAD/CAM in terms of providing practical, matured, professional and commercialised solutions (Xu *et al.* 2011). Since CAPP is a pivotal link between CAD and CAM, one of the reasons why CAPP not reaching a satisfactory level lies at the integration problems between CAPP and CAD, CAM. The proprietary data formats used on various CAD/CAM systems need corresponding information sharing methods. It is a common problem shared across the whole of manufacturing industry. Another aspect holding CAPP back is the complexity nature of planning task, which is such a complicated engineering problem. Although for each sub-task (tool selection, cutting path planning etc.) of process planning, there is a specific clear objective, 30 years research has proven that it is far more complex than it looks like. It is understandable seeing that process planning is a highly non-linear planning task with subtasks cross-coupling with each other while each subtask is rich in novel computational and reasoning problems (Bourne *et al.* 2011). Another aspect exacerbated the process planning task is the differences lying between different machining technologies, such turning, milling and turn-mill. The

optimal solution varies depending on different circumstances. That's why there are many different algorithms and implementation methods coming out of CAPP applications, including feature-based technologies (Márkus *et al.* 1997; Wang *et al.* 2006), knowledge-based system (Anwer and Chep 1999), neural network (Ming and Mak 2000), genetic algorithm (Ding *et al.* 2005), agent-based technology (Wang and Shen 2003) etc.

Of the two types of CAPP approaches, the variant approach continues to be used by some manufacturing companies. The quality of developed process plan is still dependent on the experience and knowledge background of process planner (Alting 1989). Thus the trend is toward a generative approach (Xu *et al.* 2011).

The problem with the interface between a CAPP system and a CAD system, for a generative process planning system, is that it is difficult to obtain useable features from the design data of the part. There are basically two approaches to obtain the features: design-by-feature (feature-based modelling) and feature recognition (Shah and Mantyla 1995). It is worthy to note that there are mainly three different kinds of features: functional feature, form (or geometric) feature and manufacturing (or application) feature. Design-by-feature use form feature or manufacturing feature (Maropoulos 1995b). If form feature used, there is a need to convert form feature to manufacturing feature as CAPP is standing on the viewpoint of manufacturing. Another problem with design-by-feature is that it will constrain the creativity and certainly this effect can be more pronounced when using only manufacturing features. Hence feature recognition became an essential technique for CAPP research. A short review on feature recognition is conducted and shortcomings of feature recognition methods are presented in the next section (4.3).

The interface issue between the CAPP system and CAM system is less reported than the CAD/CAPP interface. In practical, the integration problem between them can be solved by embedding process planning functions into CAM systems, like CATIA®, Pro/Engineer®, NX® etc. That is why in this thesis or in other literature the term CAD/CAM/CNC is used without CAPP. However, the CAPP functions in these systems are in many ways less structured and non-systematic. This has hindered sweeping changes to the CAPP functions and the technology up-take has only been patchy and localised (Xu *et al.* 2011).

Apart from these problems mentioned above, CAPP faces additional up-to-date challenges from the new demands and technologies advancement in product

development practice. The emerging changes pose new contents to the process planning scope. For example, the emergence of concepts of reconfigurable manufacturing and reconfigurable process plan, which calls for reconfiguration to both the hardware and software of manufacturing system, presents new challenges to process planning activity (ElMaraghy 2007).

In summary, CAPP is essential to modern CNC manufacturing. It is a long-pursuing and, despite the progresses made in this field, it is still a not fully achieved goal. With the new and deep understanding of the process planning task, it turns out to be harder than envisioned in the early days of this field (Bourne et al. 2011). It makes it clear that the current used process plans are a quite valuable resource for manufacturing industries. To take advantage of proven process plans and apply them in new products is quite important and helpful for commercial manufacturing enterprise to stay competitive in today's global market. According to the study of the PDMA (Product Development and Management Association), more than 50% of their current sales of successful high technology firms were coming from new products. In the case of the most successful overall firm, this figure was up to over 60% (Balbontin *et al.* 2000).

4.3. Feature-based technology and feature recognition methods

Feature recognition is a long evolved concept and a traditional hot spot for academic research. It is an enabling technology for CAPP system since it is mostly used to recognise features from design model. This section will give a brief introduction of feature technology and a review of state-of the-art feature recognition methods to clarify the difference and provide foundations for this research.

4.3.1. Feature technology

Feature technology is a huge concept that has been used in many different areas in various manners. Feature technology dates back into 1970's when it became a major research theme. Since then, feature technology became a huge comprehensive concept and has been used in many different areas and applications. It results into many classifications of features, such as design features, manufacturing features, inspection features, cost features etc. Hence, it is difficult to give a simple concise definition to feature though many researchers gave their definition for feature according to the specific engineering significance. Allen (2003) summarised different features definitions

and classified then into four groups: functional feature, structural feature, physical feature and form.

Features encapsulate the engineering significance of product geometry and the reasoning regarding the product in a variety of applications (Ding 2003). It can facilitate the designer with ready-to-use modules to easily express their intent. A feature database allows the reasoning system to perform some engineering analysis tasks. The information contained in the feature provides the basis for intelligent process planning and manufacturing (Chung *et al.* 1988; Rosso Jr 2005). Many applications (design by feature CAD/CAM) and even standards (STEP, STEP-NC) employ a feature-based approach. Features have been used as a major geometry and radiating element in CAD/CAM/CNC process chain. However, a feature-based system is heavily dependent on the techniques of creating a feature-based model. Feature recognition became an important topic and feature creation methods in feature technology since the seminal work of Kyprianou (1980; Han *et al.* 2000). The next section of chapter reviews the works relating to feature recognition.

4.3.2. Feature recognition methods

Since the solid modelling technique introduced in 1970s (Shah and Mantyla 1995), the field has developed a variety of techniques for unambiguous computer representations of three-dimensional objects have emerged. It results in a proliferation of sophisticated three-dimensional CAD systems widely used in manufacturing (Han *et al.* 2000). Most of feature recognition research is based on 3D solid model. In fact, feature recognition research from two-dimensional drawings can be realised based on the research on reconstruction of objects from orthographic projections (Shah and Mantyla 1995; Wesley and Markowsky 1981). Feature recognition, or feature extraction, extracts the feature at the appropriate level of abstraction from a given part model, which is suitable for the task requirements (Kailash and B. 2000). It focuses on the design and implementation of algorithms for detecting manufacturing information from solid models created by computer-aided design (CAD) systems (Han *et al.* 2000).

With feature recognition, the recognition process can be specific and optimised to a certain application (Laakko 1993) due to the non-exclusive results. The use of the feature recognition approach for creating feature-based models enables the designer to work well with the traditional CAD system without being restricted to a limited set of pre-

defined features. Furthermore, it is possible to make the design information stored in a neutral format such as STEP AP 224 (Bhandarkar and Nagi 2000).

Even today, lots of feature recognition is done manually. However, this research relates to automatic feature recognition from part programmes, and this section only focuses on automatic feature recognition methods. According to the principle and different design models evolved in the recognition process, there have been many different recognition methods, which are outlined below. However, these methods do not provide a complete solution to the problems in machining area. They are facing various criticisms including computational complexity, not efficient to deal with interacting features, multiple interpretations and machinability.

4.3.2.1. Graph matching method

“Since boundary representation (B-rep) data structures can be viewed as graph structures, graph matching has been a popular method for feature recognition(Shah and Mantyla 1995).” The graph matching method represents a B-rep model into a stereotypical sub-graph structure where the boundary element (faces, edges, vertexes) are considered nodes of the graph and the relationships between them form the arcs of the graph. The earliest work on the graph pattern analysis method was performed by Joshi (1988). The graph-based is an effective way and has its advantage over others due to the graph nature of B-rep-based solid model (Lam and Wong 2000). However, there are a number of significant drawbacks, such as large computational expenditure of sub-graph isomorphism, deficiency of dealing with interacting features and no insurance of manufacturability of the recognised feature (Han *et al.* 2000; Ding 2003).

4.3.2.2. Volume decomposition approach

The volume decomposition approach computes the removal volume from the solid model and decomposes the volume into cells for the machining purpose(Lam and Wong 2000). As the volume decomposition algorithm operates more directly on three-dimensional volumes, it is effective in handling interacting features. Wang and Kim (1998) together with Woo and Sakurai (2002) have adopted this approach, but it still suffers from expensive computations.

4.3.2.3. Rule-based approach

The Rule-based approach (Donaldson and Corney 1993) takes advantage of artificial intelligence to formalise the features based on templates consisting of a pattern of rules. This was later developed into hint-based approach (Miao *et al.* 2002). Henderson (1984) and Choi (1984) have adopted this method. However, it is not an outstanding popular approach to do feature recognition for it is impossible to define the complete rules for all the features and it cannot afford massive adjustment of features (Lam and Wong 2000).

4.3.2.4. Neural network-based method

A number of researchers have proposed artificial neural networks in feature recognition (Prabhakar 1992; Nezis 1997). A neural network consists of nodes and connections, whose generalization and learning ability of neural networks can tolerate the varying information of a solid model in the process of recognition (Lam and Wong 2000). However, for each type of feature, a neural network should be created by training using a feature definition language with sufficient input vectors defined.

4.3.2.5. Hybrid approaches

In order to take the advantage of several basic recognition methods, researchers resort to hybrid approaches to combine different basic techniques (Shah and Mantyla 1995). To solve the interacting features, Vandenbrande and Requicha (1993) proposed a hybrid method to combine the hint-based method and volume decomposition method to overcome the problem with interaction features. Gao and Shah (1998) presented another hybrid method named MCSG-based approach combining hint-based method with the conventional graph matching method. The overall algorithm is shown in Figure 4.1 where the MCSG worked as the hint. Graph matching, volume decomposition and hint-based methods are three basic and unique approaches in feature recognition. They all have their advantages and limitations. It is very difficult for each of them to be applicable to all cases. However, a hybrid method adopting selective characteristics of these approaches will provide a constructive and practical solution.

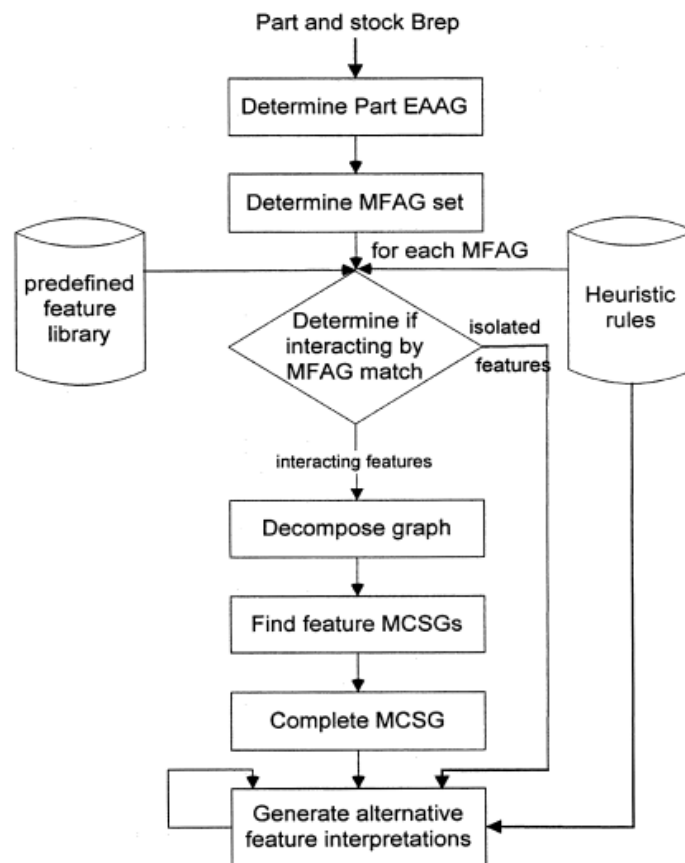


Figure 4.1 - Overview of the MCSG-based hybrid feature recognition algorithm (Gao and Shah 1998)

4.4. Knowledge management in manufacturing

The following section reviews the literature on manufacturing knowledge reuse. As knowledge management is a broad across-domain subject, this research only focuses on the manufacturing knowledge management issues. The state-of-the-art of manufacturing knowledge reuse has been presented to identify the current research gap on shopfloor knowledge capture and reuse.

4.4.1. Background of knowledge management

In modern globalised knowledge driven economic environment, previous data, information and knowledge are important commodities for organisations (Hicks *et al.* 2002). "The effective utilisation of these 'commodities' is increasingly the only way for organisations to achieve and sustain competitive advantage." Commercial companies can benefit from knowledge reuse for several reasons (Baxter *et al.* 2008): (a) companies know the product well, so are able to produce high-quality reusable knowledge; (b) the

next generation product is based on the previous version and likely to have a significant overlap with the old version. This trend will become more and more common with the move towards mass customisation with a range of product variants (Zheng *et al.* 2008). Efficient reuse of previous knowledge can be helpful to ensure the quality of the new product and reduce the lead-time of the development of the new product; and (c) knowledge reuse saves more time for innovation, since innovation is important for companies to maintain or take over competitive advantage, which is difficult to achieve, especially in mature domains.

A significant body of research work on knowledge management has addressed the subject of what is knowledge and how knowledge is related to data and information. The intuitive assumption that knowledge is something more than information has led many authors to make distinctions between data, information and knowledge (Tuomi 1999). The traditional view (Wiig 1993; Nonaka and Takeuchi 1995; Davenport and Prusak 1997; Cochrane *et al.* 2008) of the difference is to see them become more senior and contains more meaning from data to knowledge. Data is just a description of the facts, while information exists when the relationship or relevance between the data is recognised (Davenport and Prusak 1997). Knowledge is higher than information and consists of truths, judgments and know-how (Wiig 1993). It exists when commitment and belief created from the information (Nonaka and Takeuchi 1995). This classic data, information and knowledge hierarchy has been questioned by Tuomi (1999). A reversed hierarchy of data, information and knowledge is presented.

In this thesis, the discussion of the relationship between data, information and knowledge is out of the scope. The real situation might be the combination of the both arguments. Actually, besides the hierarchy of knowledge, there are different classifications: explicit and tacit (Zhou 2004; Alizon *et al.* 2006), or operational and strategic knowledge (Tissen *et al.* 1998), operational and specific knowledge (Garetti *et al.* 2005) etc. Although there are different classifications and disputations about the relationship between data, information and knowledge, all of the research identifies that knowledge is important to organisations. The aim of these discussions is managing to clarify the origin of knowledge and provide different approaches in developing information systems for knowledge management (Tuomi 1999). According to Baxter *et al.* (2007), knowledge is “actionable information” and can be stored in computer based systems in various formats. In this research, the manufacturing knowledge mainly refers

to the process information used to manufacture parts using CNC machines. It lies in tons of CNC part programmes. The author believes that the process information is technical know-how, which is valuable knowledge of production engineers. Hence, it is beyond the debate of the relationship between data, information and knowledge.

4.4.2. State-of-the-art in knowledge management in manufacturing industry

Efficient management of product information is critical to the enhancement of corporate competitiveness (Kim *et al.* 2001). Product data management (PDM) system became available for that purpose. It integrates and manages all applications, information, and processes that define a product, from design to manufacture, and to end-user support (Liu and Xu 2001). PDM focuses primarily on engineering tasks and doesn't offer a complete data view over the entire product life cycle. Later, a new generation of software systems, Product Lifecycle Management (PLM) systems, evolved from the PDM systems (Cheung and Schaefer 2009). Portella (2002) described PLM systems as "an infrastructure to support management of product definition throughout its complete lifecycle from initial concept to product obsolescence". As computer aided manufacturing is a component of PLM, PLM systems are a popular solution for managing the integrated Knowledge Information and Data regarding product design, production process (Denkena *et al.* 2007). Denkena *et al.* proposed a holistic PLM/CAPP solution for knowledge management for process planning. A complementary set of tools and capabilities required for providing a holistic PLM/CAPP solution has been specified.

Garetti *et al.* (2005) debated the role of knowledge management in PLM projects. There were two types knowledge in companies: operational and specific. The specific knowledge deals with particular products and processes related to the know-how used in the making processes. They concluded that PLM is quite suitable to manage company knowledge. The new trend of the development of PLM towards web oriented structure has been enabling many companies to spread their know-how in an easier way and considerably contributed to the enlarged business concept. Regarding the specific knowledge, the authors also suggest using interviews with experts to capture the knowledge and store it into the databases of the knowledge management tools.

Chen *et al.* (2008) proposed an ANP model with sensitivity analysis for new product development method selection. Together with the model, PLM with suitable knowledge management methods and process development management, were adopted to ensure

the successful execution of product development process. An anonymous manufacturing firm was used to validate the model. The performances of the investigated firm before and after using PLM were compared to demonstrate the effectiveness of the models.

Instead of focusing on the whole product life or company level knowledge management, considerable research has been carried out on particular stages of product life: design, process planning, production, assembly and services etc.

Reed et al. (2011) investigated the role of knowledge based systems for design reuse. This research came to the conclusion that knowledge based system can be a primary source of codified knowledge for designers and successfully reduce the reliance of the business on critical knowledge holders. It can be vital viable alternative to verbal knowledge exchange. It is recommended that knowledge capture activities should be established as a recognised and necessary part of knowledge management projects for long term success.

Baxter et al. (2007) investigated the design knowledge reuse method using a process modelling approach. The authors argued that current design knowledge reuse systems focus exclusively on geometrical data and miss non-geometric knowledge elements. The proposed methodology provides an integrated design knowledge reuse framework including a Process Model and a Product Model. A case study was presented using this approach. Then this knowledge reuse framework was expanded to include requirements management (Baxter *et al.* 2008).

Zhang and Zhou (2011) recognised the importance to reuse existing designs to reduce the cost and shorten the cycle of new product development. An effective system to manage and reuse existing design resources was established. A novel two-step algorithm was used to analyse the similarity and reusability of product design resources. For the same purpose, Mun and Ramani (2011) provided a method using ontology and multi-criteria decision making (MCDM) technique to measure the similarity between parts' specifications in a database and the query data.

Zheng et al. (2008) presented a process knowledge reuse approach for rapid process configuration. A systematic knowledge model with six different levels of granularity to represent process knowledge was proposed: (i) core process skeletons, (ii) process networks, (iii) process routes, (iv) process segments, (v) processes/workplan, and (vi)

operations/working step. Using the knowledge model a prototype entitled Visual Process Planning (VISPP) is implemented, which is a graphical and flowchart style software tool for rapid process configuration. A case study from the aerospace industry has been used to verify the proposed process knowledge model and process configuration methodology.

To reuse the manufacturing knowledge in decision support systems, Cochrane et al. (2008) gave three principles: the separation of information from knowledge, the separation of product knowledge from manufacturing process knowledge, and the correct classification of manufacturing knowledge. The key contribution of this research is the guidelines presented to provide improved guidance on how to classify manufacturing knowledge for optimum reuse. A manufacturability analysis platform (MAP) has been used to test and evaluate the principles.

Alizon et al. (2006) proposed to reuse manufacturing knowledge to facilitate platform-based production. An investigation on how to reuse manufacturing information based on process modelling techniques with the product platform technology has been carried out. Product platform is a technique for exploiting commonality across a family of products. The similarity of process information provides an opportunity to reuse existing experience to develop new products. A Reuse Existing Unit for Shape and Efficiency (R.E.U.S.E.) method has been proposed to facilitate the search and reuse of information in a repository through three stages that consider similarity, efficiency, and configuration. A case study of an assembly plant was carried out to show the potential implementation of the method.

Researchers (Liu *et al.* 2007; Guo *et al.* 2011; Schroeder and Hoffmann 2006) and commercial companies (CGTech 2012; Predator Software 2012) who worked on part programme translation provides another solution for process knowledge reuse from proven NC programmes. It is a good practice only under the premise that the toolpath is suitable and can be performed with another machine and its tools. The scenario is quite limited. More importantly, the process knowledge cannot be borrowed into other products with different manufacturing resources. The problem lies at the binding of product and process information.

Although knowledge management in manufacturing industries is a long recognised issue, the knowledge management activities including capturing, maintenance and sharing are

still inefficiently solved due to various problems such as incomplete and/or ill-structured knowledge, obsoleted knowledge under poor maintenance or failed continuous update and horizontal Communities of Practice (e.g. over design departments of large companies). The vertically linkages (e.g. between planning and shop-floor areas, including employees with different levels of expertise etc.) are rare or unidirectional (Fischer and Stokic 2002). Actually, for industrial enterprises, the shopfloor is an important and knowledge intensive stage. Most of the employees are working at or related with the shop floor. According to a knowledge management study by Mittelman (2005), there are two thirds of 7000 employees are blue-collar workers in the steel division in Linz. Critical knowledge and know-how for the production processes is located in the brains of the workers. Knowledge management is not practical and reasonable without considering the shopfloor.

Considering the shopfloor of CNC manufacturing, it is the final stage where real material removal happens. Since manufacturing activities from design to production of the final part are performed sequentially, the detailed process plan has been already determined before it comes to the shopfloor. This process plan is generated based on the shopfloor status information. In current process planning practice, the information is static while the shopfloor itself is a complex and ever-changing environment. Therefore, the process plan and the resulting part programme are not based on live data and might not be optimum for the actual available shopfloor resources. Consequently, sometimes, the operators will change some parameters according to their experience or familiarity with the actual resources. From this viewpoint, part programmes are not only the final but also the most accurate representation of a process plan that is used to make the physical part. Additionally, with the introduction of feature based technology, shopfloor CAD/CAM systems (Shopmill, Shopturn) enables people make parts easily. A part can be designed, planned and machined only at shopfloor. The part programmes are the only records. Furthermore, in the situation of parts without using CAD/CAM system (legacy parts) or CAD/CAM file lost, the part programmes are even more valuable.

Although the shopfloor knowledge is valuable for the enterprise, and can be used to improve quality and reduce the cost and cycle time of products, it is unlikely to be available at the design and process planning stages before the generation of part programmes. Due to the lack of interoperability, there is limited opportunity to feed this

knowledge back through the CAx chain, especially in an automatic manner (Newman *et al.* 2008).

4.5. Critique

According to the review on CAPP and enabling technology feature recognition, although various feature recognition research efforts have been proposed, this key technology linking CAD and CAM is infancy and still requires intensive efforts to make it suitable for actual industrial use (Vermaa and Rajotiab 2010). Since the CAPP systems is not so sophisticated as other CAx systems, it makes the existing process knowledge in part programmes even more valuable and worthy to capture and reuse. Based on the review in knowledge management, the following research gaps have been identified:

4.5.1. Shopfloor gap

Most of the reported work on knowledge management in manufacturing paid attention on the stages of design and process planning. Shopfloor is the last important or the sole stage of production and should be included in the knowledge management project. Part programmes are obviously the most important documents and the most accurate representation of production knowledge. The lagging behind of CAPP is another justification for the value of a proven part programme. How to manage and capture the knowledge through part programmes and how to reuse it in design and process planning stages, the vertical linkages, need more research efforts.

4.5.2. Product and process information separation gap

The current literature on knowledge reuse through part programmes has a disadvantage due to the data structure of part programming languages. The problem is the mixture of product and process information. The most widely used programming language is G&M codes. It describes tool movements and machines functions and contains process and product information together. To reuse the process knowledge, there is need to separate it from product information. That is the reason why the direct translation of part programmes cannot be implemented widely. Thus in order to realise the capture and reuse the process knowledge in a standardised and wide applicable manner, the separation between the product information and process knowledge is necessary.

4.5.3. Knowledge representation gap

To capture the process knowledge from part programmes, the representation of the knowledge is another issue. Currently there is no standardised data format for knowledge capture. Ill-structured knowledge is one reason responsible for knowledge management failure due to the problems of maintenance and sharing. Fortunately, STEP-NC has been proposed to be the data model for computerised numerical controllers. It is designed to accommodate CNC machining process information. Additionally, it is compatible with the STEP standard to incorporate the product data. While encapsulating product and process information, STEP-NC keeps them separate. Hence, the STEP-NC standardised data model can be considered as a candidate for knowledge representation. Using STEP-NC to facilitate knowledge management could be an appropriate solution for the current problems of shopfloor knowledge reuse.

In the next chapter, all these research gaps are tackled in this research, with alternative solutions identified and selected solution described in detail.

5. A framework for universal process comprehension for interoperable CNC manufacturing

5.1. Introduction

Following the critique in Chapter 3 and Chapter 4, possible solutions to interoperability problem have been provided. Based on the comparison of the alternative solutions, a universal process comprehension approach have been selected. This chapter presents the overall framework of the research to realise the interoperability between CNC machines and other manufacturing resources such as CAD/CAM systems, other CNC machines etc. Within the research context described in Chapter 2, the critique has been used to develop a set of requirements for the universal process comprehension research. These requirements have been reflected in the necessary functionalities of the framework. The functionalities, realisation methodologies together with their interconnections and workflow are then introduced and analysed.

5.2. Possible solutions

To realise the research aims, several possible solutions have been considered:

Resource to resource translators. This is the most straightforward way to reuse the knowledge and realise the flexibility. To reuse part programmes between different CNC machines, machine-to-machine specific part programme translators can be developed. For the information exchange between each CNC machine and other systems, other specific translators can be developed. Using this method, there is a need to develop lots of specific translators for each resource combination. Furthermore, the interoperability achieved in this way is quite limited. For example, the machine-to-machine convertors are translating toolpaths within the part programme, in which case the cutting tool used to perform the task should be of the same diameter.

Part programme to CAD model. Another method is utilising the shopfloor part programmes to recognise the part geometries and restore the product CAD model. Using the CAD model, other systems such as CNC machines can be programmed through CAM system. This method provides an approach to reuse the design model, but not the manufacturing process knowledge. It still needs to develop lots of machine-to-CAD convertors and capture the process knowledge manually.

Process comprehension method. Process comprehension is the appreciation and interpretation of product information and manufacturing process knowledge contained in various part programmes. It can derive both product and process knowledge from the shopfloor and make it ready to share across different systems. Since there are various programming languages/dialects used on different machines, there is a need to develop different process comprehension interfaces for each language/dialect.

Universal process comprehension method. Compared with last method, it provides a universal approach to deal with different types of part programmes while keeps the advantage of reserving both the product and process knowledge.

In this research, the universal process comprehension method has been chosen as the appropriate method to achieve the research aims.

5.3. Universal process comprehension

As CNC vendors adopt their own programming dialects, post processors are needed for each pair of CAM/CNC configuration to translate the processes logic contained in the CAM system to part programmes appropriate for a specific CNC machine. This is a downstream flow of information. The name “process comprehension” was proposed for the opposite of the above process as reverse post processing does not capture the essence of this operation since it is not just a reverse information translation. Comprehension is the capability of understanding and restoring the original information. Process comprehension is essentially restructuring the manufacturing information contained in a CNC part programme into a high-level process plan and the associated resource information. As shown in Figure 5.1 the part programme contains toolpath and machine functions. After process comprehension, the process plan is reconstructed including high level manufacturing information such as features, machining strategies etc.

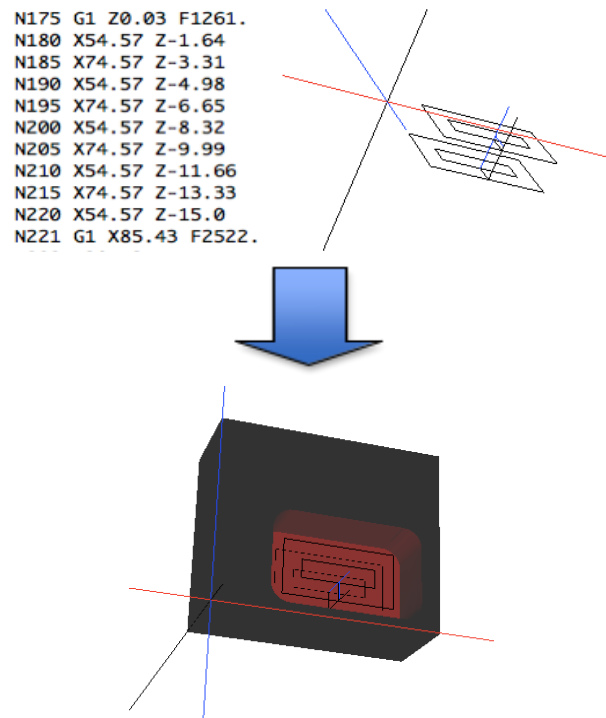


Figure 5.1 - The concept of process comprehension

In order to carry out process comprehension, sophisticated systems should be developed to convert the information. As shown in Figure 5.2, for each programming dialect, a proprietary process comprehension interface would be needed. However, the work to develop “reverse” processors for each CNC/CAM configuration is tremendous and, more importantly, not an efficient way. This situation is quite similar to post processors in terms of complexity and efficiency. That is why the solution of a universal post processor was proposed (NCCS 2011; MIS Group 2011; ESPRIT 2011). In author’s research, a universal process comprehension interface (UPCi) is proposed to support various CNC machines regardless of their programming dialects. The interface here means the communication connector between different systems. In order to realise interoperability between different resources, the output of UPCi should be in a generic and standardised manner. A standardised format means resource independent, which is essential to ensure the universal applicability of the output of UPCi on different resources.

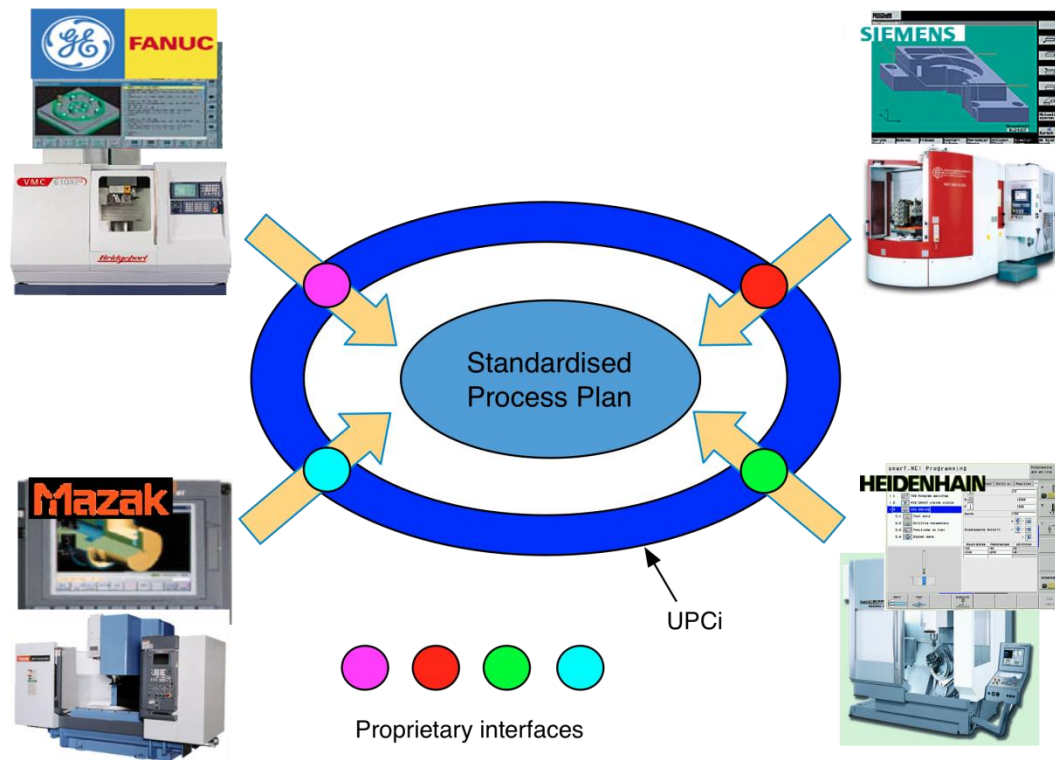


Figure 5.2 - A universal process comprehension interface

5.4. Design considerations for a universal process comprehension

The intention of UPCI is to compile manufacturing knowledge contained in CNC part programmes into a generic process plan format. To reach this goal, based on the critiques derived in Section 3.5, four considerations have to be taken into account in the development of an effective framework:

5.4.1. Minimum modification to current resources

As discussed in Section 3.5.1, STEP-NC can provide the mechanism for resources interoperability in manufacturing. However, the implementation of STEP-NC needs all the vendors of manufacturing resources to make every effort to adopt this new data model. From the view point of the leading companies of CAD, CAM, CNC machines, there are no commercial motivations. It is basically the reason why STEP-NC cannot be implemented directly. From this viewpoint, the realisation of shopfloor interoperability should avoid significant modifications of current resources to minimise the implementation resistance.

5.4.2. Universal support architecture

The universal process comprehension explored in this research should have an architecture that supports the wide range of CNC machines and processes, and ensures excellent extensibility in the future. As there are many different programming languages for CNCs, the architecture of this approach should be able to accommodate different information formats in a common style.

5.4.3. Shopfloor knowledge capture ability

Due to the unidirectional information flow of CAX chain, it is difficult to capture and feedback valuable shopfloor knowledge to the planning department in an automatic way. UPCI should be able to capture the shopfloor knowledge and make it ready to be shared across different manufacturing resources.

5.4.4. CAX interoperability

To realise CAX interoperability, the information flow among the CAX chain should be realised in an automatic manner, while keeping the integrity of the information. The semantic interpretation the information between different systems should be identical. This will be validated by machining experiments of three test parts with representative manufacturing features on different manufacturing resources.

5.5. Design of the universal process comprehension framework

Based on the design considerations developed, the framework of the universal process comprehension is proposed in this section. It serves as a blueprint to achieve the research goal specified in Chapter 2. The overall representation of the framework is introduced in Section 5.5.1. The detailed structure and key methodologies of implementation are then discussed in the following sections.

5.5.1. Functional view of universal process comprehension

The top-level functional model of universal process comprehension is shown in Figure 5.3. From this IDEF0 view, the input includes G&M codes based part programmes, workpiece geometry and cutting tool information. Universal process comprehension is used to comprehend the original process plan and compile them into STEP-NC representation.

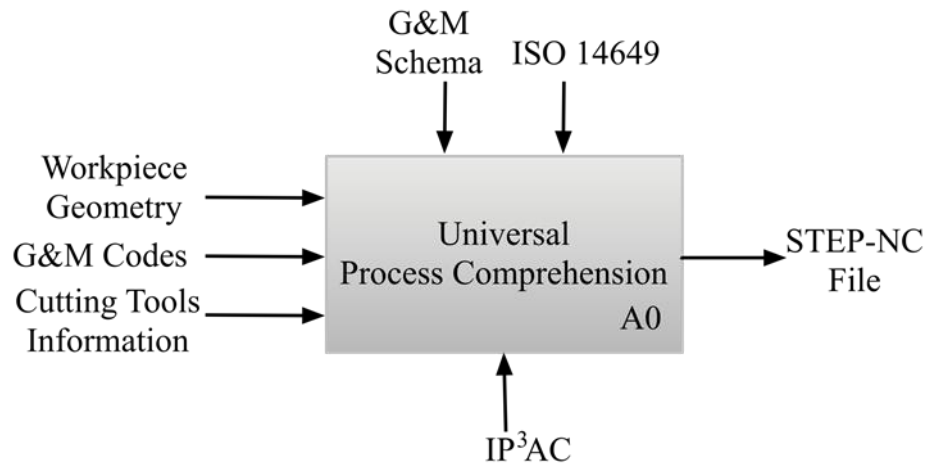


Figure 5.3 - Overview of universal process comprehension

To understand different part programming dialects, a programming specification is needed to work as a reference dictionary. For the generic output of universal process comprehension, STEP-NC, more specifically ISO 14649, has been chosen to be the representation format of process plan for three major reasons: first, STEP-NC covers comprehensive suite of manufacturing information in a concise manner. It offers necessary entities to capture the shopfloor knowledge including machining strategies, cutting parameters etc. Furthermore, STEP-NC provides the appropriate data model to represent the process plan in a resource independent manner. This is quite essential to realise the interoperability between different resources. In addition, while both ISO 14649 and AP 238 can offer these two benefits, ISO 14649 provides adequate entities to capsule product geometry information for prismatic parts, without the performance sacrifice of complex AP 238 data (Kramer *et al.* 2006).

With process information extracted from part programmes, a previously developed Java library for manipulating STEP-NC data entitled the Integrated Platform for Process Planning and Control (IP³AC) (Nassehi *et al.* 2007), has been used to manipulate and generate STEP-NC data based on the process information. When process information is expressed in such a resource independent format, it is ready to be shared with other STEP-NC compliant systems or can be processed into resource specific data if necessary.

5.5.2. Flexible computerised Meta-model of CNC activities

As discussed in Section 5.3, a universal process comprehension interface is to be developed in this research, instead of develop unique process comprehension interfaces for each dialect. In order to support the different dialects used on various CNC machines,

a computerised meta-model of CNC activities needs to be developed. It summarises and abstracts common characteristics of CNC machines activities to represent the information contained in part programmes. Consequently, the subsequent activities in process comprehension only deal with this metadata. A single set of algorithms is hence needed to realise the process comprehension without worrying about the differences between various NC programming dialects. Different programming dialects can be universally supported in the process comprehension framework by translating them into this Meta-model.

To realise the translation between CNC Meta-model and programming dialects, descriptions of their programming syntax of different dialects is needed to work as a dictionary. For each dialect, the description of its programming syntax is an entry of the dictionary. Hence this dictionary enables the process comprehension framework to cover new programming dialects by adding new entries to the dictionary. XML has been chosen to be the description language for its excellent design advantages such as being human-legible and reasonably clear, easy to process, wide support and straightforwardly usable over the Internet (W3C Recommendation 2008). For this research, using XML gives the author the ability to define the meaningful custom-tailored tags that is engineer-legible. Another advantage of XML is its characteristic of easy transportation over the internet for that the ability of sharing and exchanging information over the internet becomes the prerequisite for the newly emerged manufacturing technologies (Xu 2006b).

The mapping rule between the CNC Meta-model and syntax of a G&M code dialect (G&M schema) can be defined in an XML file and connects the CNC Meta-model and the dialect, as shown in Figure 5.4. A predesigned template of XML description of CNC dialects needs to be developed to facilitate the development of new dialects. Thus it is possible for the user of UPCI to extend it to support their specific CNC machines' specific programming languages without needing structural changes. Chapter 6 presents a detailed description of this approach to achieve the desired functionality.

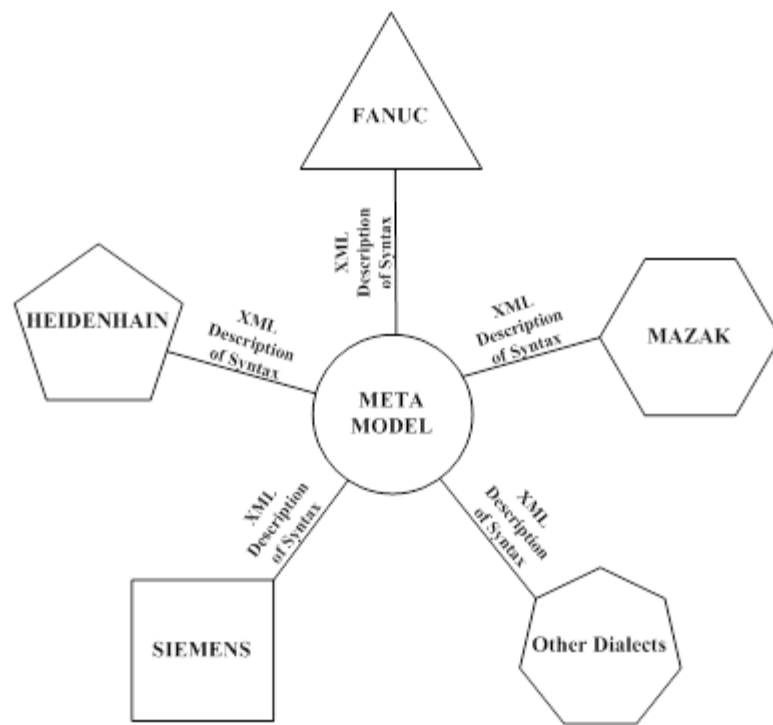


Figure 5.4 - Meta-model of CNC programming languages

5.5.3. Reconstruction of process plan in a standardised format

As STEP-NC describes the manufacturing knowledge in a feature oriented manner, the machine functions and cutting tool movements represented by the meta-model are used to generate the workingsteps, identify the machining features and create the proper machining operations. It is essential since STEP-NC has a level of feature based description. However, the complexity of the feature required is different from the “traditional” CAD or CAPP applications.

Compared with creation of workingsteps and operations, feature recognition from G&M codes is the most important and challenging task in this research context. Since feature recognition is usually applied in the design area to identify features from CAD data, in this research, it substantially different to the classic feature recognition research. To the author’s knowledge, there is no research reported on this topic for milling operations.

For feature recognition, the cutting tool information is necessary and vital. It provides significant information for feature recognition and operation generation. Based on the data translated into the meta-model, some initial data (i.e. tool change, feedrate alterations etc.) is used to divide the part programme into several sections representing

the individual operations. For each section, the coordinates of toolpath and tool radius are used to calculate the outer boundary. Then, this boundary information, together with cutting tool type and the geometry of the raw workpiece are used to identify the feature type. For each section, a feature and an operation can be generated. However, for a single feature it is common to have more than one, usually two, operations. It means it is possible for two sections to share one feature. Hence, the next step is to identify the features by the placement and geometry information, remove the duplicated features and associate two operations to one feature as roughing and finishing. This feature recognition approach will be investigated in Chapter 7.

5.5.4. Functional architecture of universal process comprehension framework

The activity view of the universal process comprehension framework is depicted as an IDEF0 diagram shown in Figure 5.5. It is a more detailed version of the IDEF0 diagram in Figure 5.3. It shows the detailed activities flow to realise the process comprehension of process plan from low-level machining information at shopfloor.

As seen in Figure 5.5, the input is the G&M code part programme for the CNC machine together with cutting tool information and raw workpiece geometry. To translate the programme into the meta-model, the CNC machine specific G&M commands' definition is needed in the form of a G&M schema. This schema is defined using an XML file template. With the use of the G&M code schemas, the syntax of the part programme is checked and the low level semantics of the machine specific commands are translated into a set of inter-related Java objects. These objects are instantiated by the Java classes of the CNC Meta-model to represent CNC machine functions and cutting tool movements.

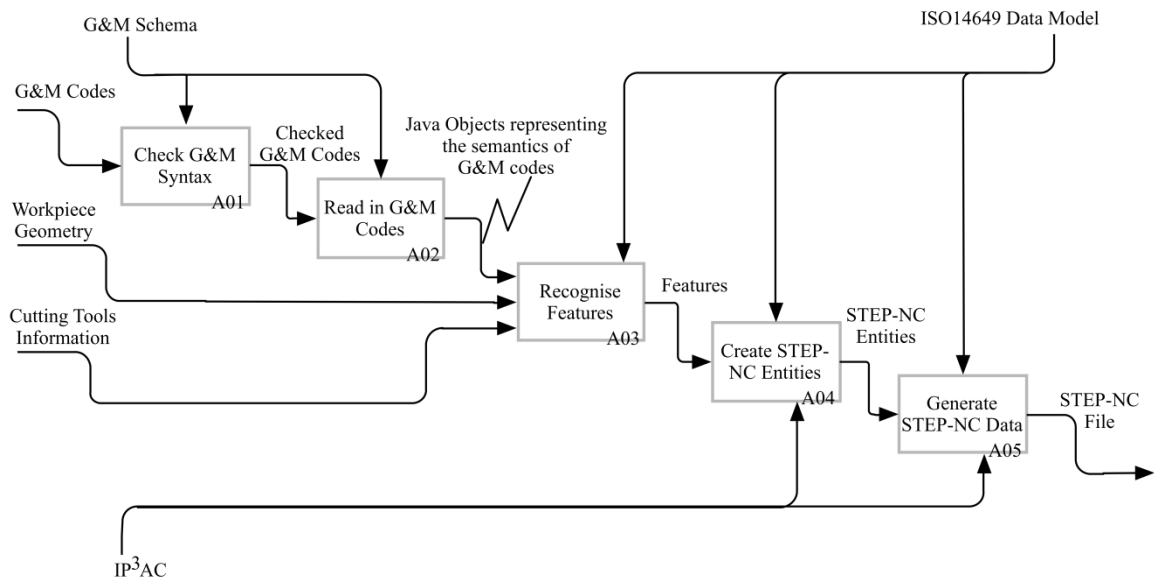


Figure 5.5 - IDEF0 representation of universal process comprehension framework

The next stage is to derive the original process plan of the part programme by recognising the manufacturing features and generate the STEP-NC entities using IP³AC to represent the process plan information. For successful feature recognition, apart from part programmes, cutting tool information and workpiece geometry are also needed at this stage. After the feature recognition, the associated operation attributes such as spindle speed, feedrate, coolant etc. are reconstructed. After this stage, these features and operations are organised by STEP-NC entities: project, workplan and workingstep etc. The standard representation of the process plan for the part can then be generated and can be shared across different manufacturing systems, as shown in Figure 5.6. CNC machines at the shopfloor are connected to other resources and shopfloor knowledge can be captured and reused in other systems. For instance, this standardised process plan generated from UP³i can be used in a CAM system to regenerate part programmes for a legacy CNC machine tool. An interoperable manufacturing network connected with shopfloor can be realised through universal process comprehension.

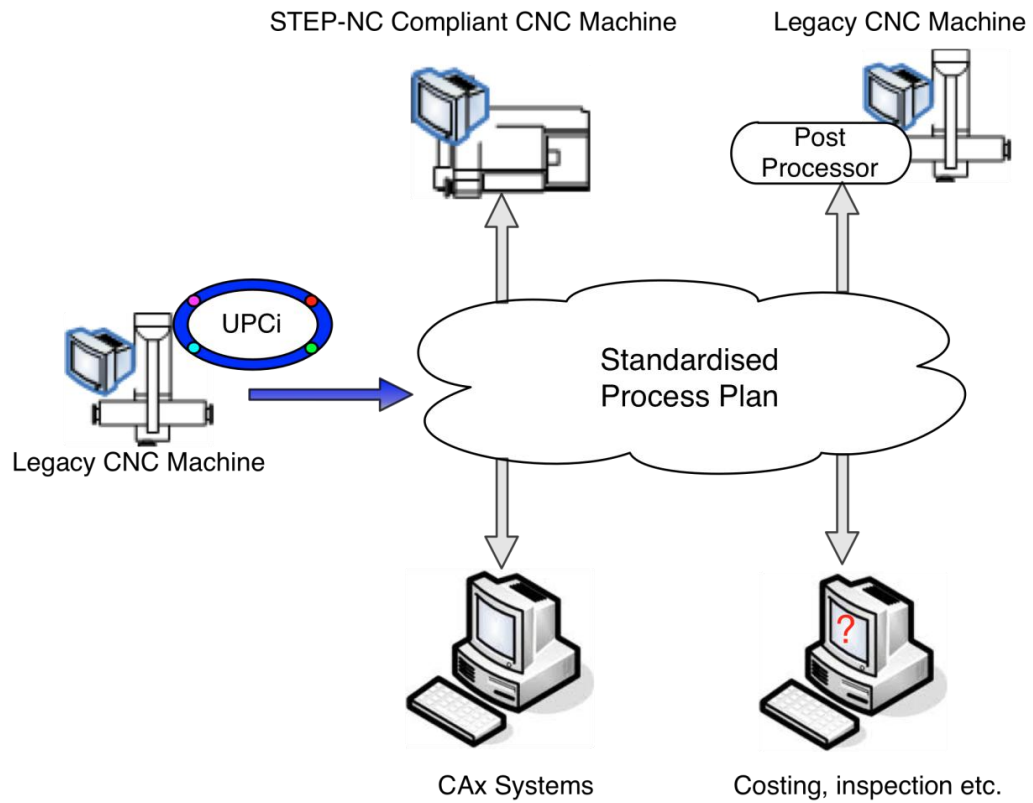


Figure 5.6 - Interoperable manufacturing enabled by UPCi

5.6. Summary

This chapter has highlighted the framework of the universal process comprehension research. Based on the design considerations, a design of the framework has been proposed. The fundamental aspects and the implementation methodologies have been provided. The following three chapters will focus on several specific implementing techniques of the framework. Chapter 6 proposes a Meta-model of CNC programming languages. Chapter 7 researches on the feature and operation recognition method. The software prototype implementation of UPCi is introduced in Chapter 8.

6. A meta-model of CNC programming languages

6.1. Introduction

In this chapter, the possibility to interpret CNC part programmes via a single type of process comprehension interface is explored by comparing different CNC programming dialects. A computerised Meta-model of CNC programming languages is proposed. This model is devised with the aim to be a neutral equivalent of different CNC programming languages to represent the process plan in part programmes. With CNC commands interpreted in this model, manufacturing information contained in part programmes can be accessed via a single type of interface without the concern of the specific syntax or semantics supported by the CNC machine on which the part programmes is used. An XML method to translate CNC dialects to this model is presented.

6.2. Comparison of part programming systems on modern CNC machines

As discussed in Chapter 2, G&M codes are widely used in today's commercial CNC controllers, which makes it de-facto standard for CNC programming. However, CNC controller developers have extended this standard with their own supplementary commands to programme some machine specific features. Some of them even develop new programming languages for their own CNC machines (Mazak 2010; Heidenhain 2009). The result is that there are massive number of CNC programming dialects used in manufacturing industries.

From the semantics point of view, despite the different formats of part programming languages, in essence they are the same. They all are the motion commands for physical machine tools, which can be equipped with a controller of any make. That is the reason why identical or similar machine tools, such as 3-axis vertical CNC milling machines, can be controlled by either Fanuc or Siemens controllers. For the same manufactured part, identical machine tools equipped with different controllers perform the similar or even identical motions with different part programs to machine it. Thus, the real difference between programming languages is their presentations and interpreting methods. The logic and micro-process contained in these programmes are the same. This can be substantiated from the mechanism of the CAM system. In a CAM system, after the process planner defines the process plan, the micro-process data including tool path and

associated switching operations such as tool change, coolant on/off etc. will be calculated and generated. The next step is to choose the right post-processor to convert this data to the part programme for the corresponding CNC machine, as shown in Figure 6.1. The process plan (cutter location data, machine functions etc.) in the CAM system is resource independent, and theoretically, can be converted to any part programmes provided there is a suitable post processor available and the controller is capable to perform the task. A part programme that is post processed becomes resource dependent; in other words, it will be tied to a specific combination of machine tool and controller as defined in the post processor. From this point of view, part programmes are the machine specific interpretation of the underlying process plan. Theoretically, it is possible to convert different part programmes back to a resource independent representation.

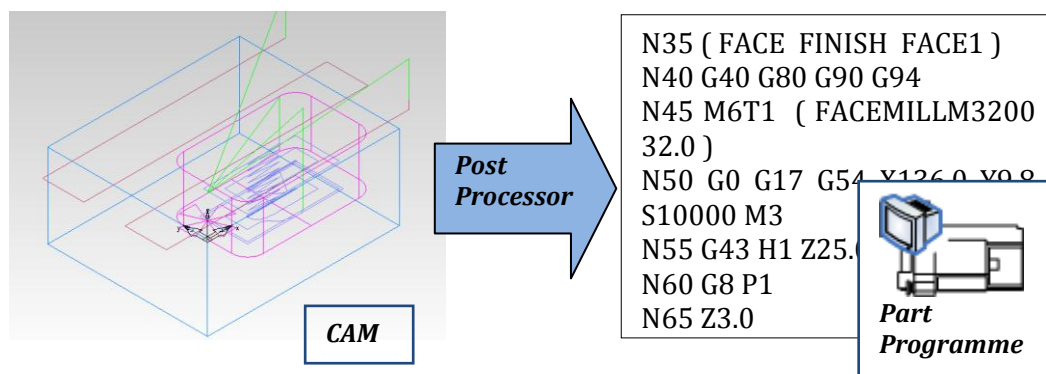


Figure 6.1 - Process plan in CAM to resource dependent part programmes

From the syntax presentation of the programming languages, it is possible to translate the part programme to a neutral language. For example in Table 6.1 (GE Fanuc Automation 1998; Siemens AG 2000; Heidenhain 2009), there are several commands format for a Fanuc controller, a Siemens controller and a Heidenhain controller. For a linear Interpolation (straight cutting travel of the cutting tool from one position to another specified by the coordinates), Fanuc uses G1 and Heidenhain uses L. Similarly, for clockwise circular interpolation commands in XY-plane, Fanuc has two type of commands format by specifying the centre of the arc and the radius. Whilst the Siemens controller has two more command formats to identify the same arc toolpath: by specifying the opening angle or an intermediate point. Heidenhain controllers can be programmed using their proprietary language and have a special format for an arc path using command "CT" without defining centre and radius, as shown in the table. The arc path starts from last position at a tangent with the previously programmed contour

element and ends with the current position. Although they use different command formats, the semantics behind them is the same. In this comparison table, they are ordering the cutting tool to move along a straight line, and an arc or drilling hole with chip breaking movements. Hence, if a neutral data model is available, different part programmes can be translate into a neutral representation, provided a translation dictionary is available. In fact, there is such a neutral representation developed by NIST (Proctor *et al.* 1997).

Table 6.1 - Comparisons between Fanuc, Siemens and Heidenhain programming dialects

Commands Controller	Fanuc	Siemens	Heidenhain
Linear Interpolation	G01 X..Y..	G1 X..Y..	L X+..Y+..
Clockwise Circular Interpolation	G02 X..Y.. I..J.. <i>or</i> G02 X..Y..R..	G2 X..Y..I..J.. <i>or</i> G2 X..Y..CR=.. <i>or</i> G2 X..Y..AR=.. <i>or</i> CIP X..Y..I1=..J1=..	C X+..Y+..DR.. <i>or</i> CR X+..Y+..R+.. <i>or</i> CT X+..Y+..
Peck Drilling Cycle	G83...	CYCLE 83...	Cycle 1...
Input Unit	G20 – Inch G21 – Metric	G70 – Inch G71 – Metric	INCH MM

Notes:

1. Examples of NC commands in this table are applied for XY plane.
2. The values of the parameters are ignored and indicted by “..”.
3. Peck Drilling cycle parameters are complex and not included in this table.

The neutral data model proposed by NIST called Canonical Machining Commands to represent various programming dialects (Proctor *et al.* 1997). The aim of the canonical commands is to represent RS274 (US equivalent of ISO 6983) codes to realise one-to-one correspondences with commands employed by commercial motion control boards. However, this model is designed for the RS274 language dialects and it embraces more specific and low-level data. For example, a peck drilling cycle in CNC commands can be decomposed into several, possibly dozens of, canonical commands. For this research, micro-process information contained in CNC part programmes is already low-level information. There is no need to interpret this low-level information into a lower and more specific level. On the contrary, it is better to keep the information granularity to help to get a higher-level understanding of the process data. It is one of the fundamental goals of this research as specified in Chapter 2. Consequently, a new Meta-model of CNC

programming languages is proposed to represent the process information contained in part programmes written in various programming dialects.

6.3. A meta-model of CNC programming languages

The aim of the Meta-model is to uniform the following process comprehension algorithms regardless the differences between programming dialects. As shown in Figure 6.2, if there is no such a model, there is a need to develop separate comprehension interfaces for each type of CNC programming dialects. This work should be huge and not practical for implementation since there are more than 5000 programming dialects in existence (Maeder *et al.* 2002; STEP Tools Inc 2012). With this Meta-model, different dialects can be translated into this model and only one standardised process comprehension interface (universal process comprehension) is needed regardless of the type of the languages. It significantly simplifies the task of the development of process comprehension algorithms. Also, The modularisation, to have separate modules of inputting and handling data, helps to make the system less complex and easy to test.

This model has been devised with three objectives. First, the Meta-model should cover general functionalities of common 3-axis CNC milling machine tools. Second, it had to be able to interpret part programmes based on ISO 6983 commands and had mechanisms to incorporate other proprietary programming languages. Third, it should be able to represent the original process information and keep its integrity, granularity and homogeneity.

In this research a preliminary version of the Meta-model has been developed. This version covers most of the motion commands and some miscellaneous settings. Four types of frequently used drilling cycles are included. A list of the Meta-model of CNC activities is shown in Table 6.2.

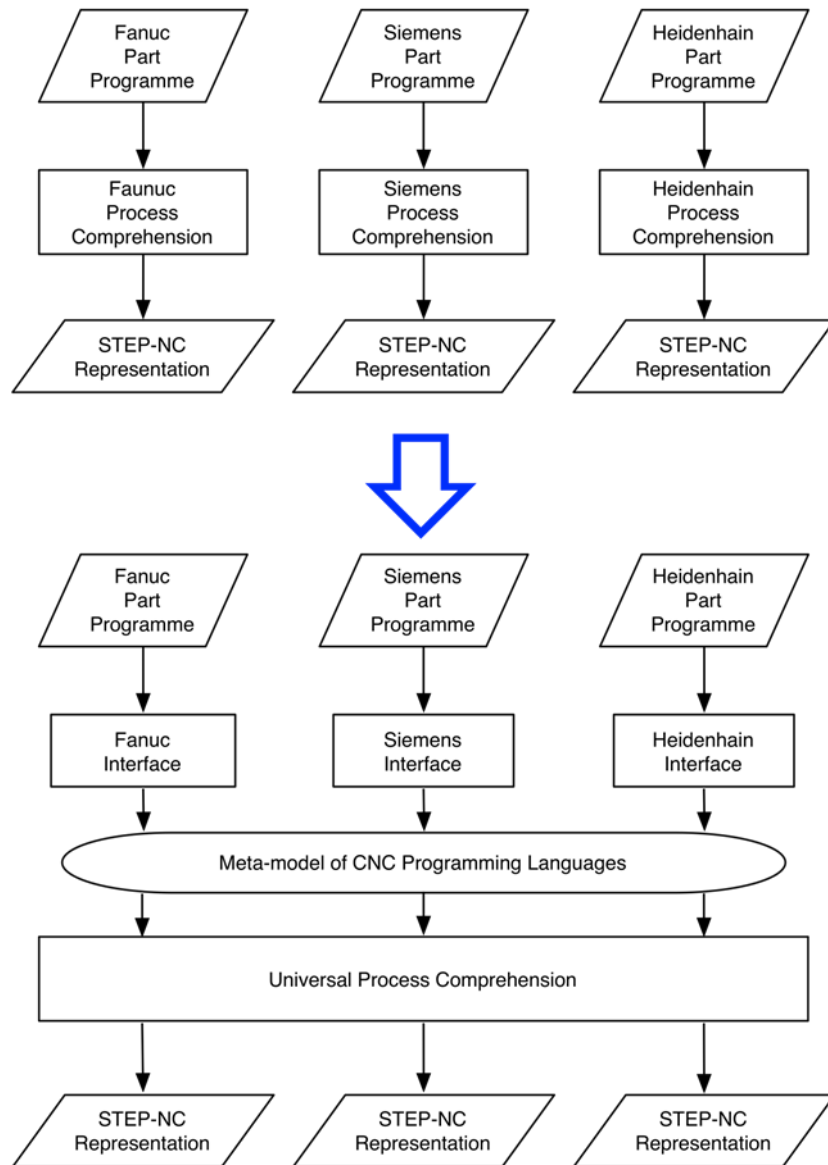


Figure 6.2 - Meta-model used for universal process comprehension

The Meta-model entities listed in the table are developed based on the most widely used G&M codes and it meets those three objectives specified earlier. It covers most frequently used G&M commands. However it is not restricted with G&M codes. Judging from the names of these entities and their descriptions, it covers common movements and settings of all kinds of CNC machines regardless what kind of controllers are equipped on them. Hence it is possible to translate other CNC programming languages other than G&M based dialects into this model. Consequently it can represent the process information contained in part programmes. Furthermore, it keeps and in some cases enhances the information granularity. For example, there are separate commands to specify the plane

or measure units in G&M codes. In the Meta-model, there is only one entity to model these choices.

Table 6.2 - Meta-model entities of CNC activities

<i>Meta-model entities</i>	<i>Functions</i>
RepaidPosition	Repaid positioning.
LinearInterpolation	Linear movement of cutting tool.
CircularInterpolation	Circular movement of cutting tool.
Drilling	Drilling cycle or spot drilling cycle to create a common hole or a guide hole.
Boring	Boring cycle to enlarge an already existing hole using a single point cutter.
PeckDrilling	Multi-step drilling cycle to create a deep hole.
CounterBoring	Drilling cycle to create a stepped hole.
CycleCancel	Cancel canned cycle in modal (continuous effect).
CommandManner	Absolute or Incremental command.
PlaneSelection	Plane selection for circular interpolation.
Unit	Measure unit: MM/INCH.
ToolFunction	Tool call and tool change.
Feedrate	Feedrate.
Spindle	Spindle ON/OFF switch and spindle speed.
Coolant	Coolant ON/OFF.
End	Programme stop/end.

6.4. Modelling NC languages in XML description

With the Meta-model of CNC programming languages, various CNC dialects can be handled by a single standardised process comprehension interface. It simplifies the task of process comprehension significantly since there is no need to develop different algorithms for each CNC dialect. However, there is a need to develop translation interface between a CNC dialect and the Meta-model. A traditional and straightforward way to do this is developing proprietary interfaces to read in different dialects, as shown in Figure 6.3. It means there is need to develop lots of this kind of interfaces in advance to support

different dialects. In fact it is possible to use a “dictionary” mechanism to standardised theses translation interfaces, as shown in Figure 6.4.

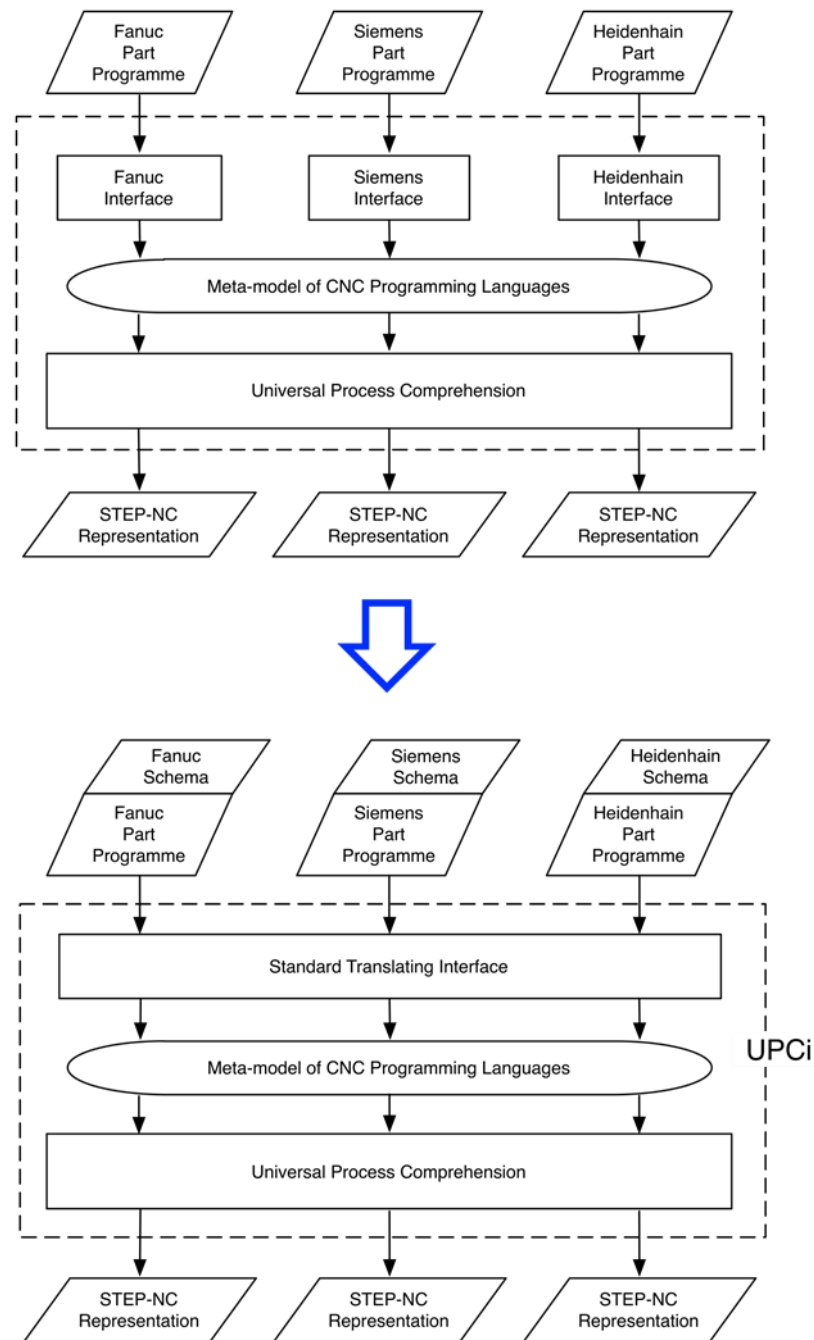


Figure 6.3 - Meta-model used for standardised process comprehension

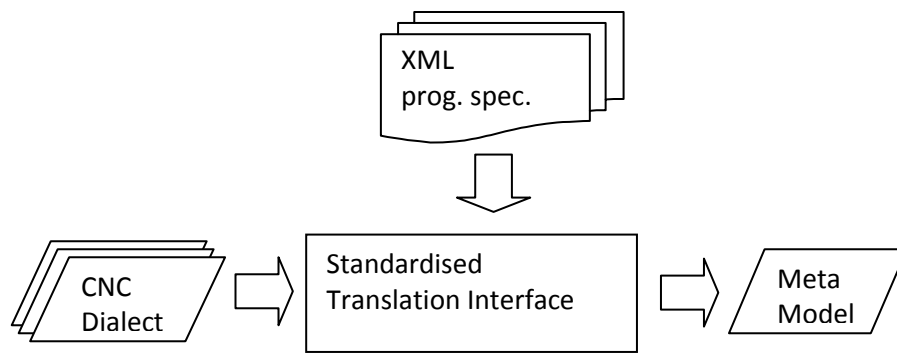


Figure 6.4 - Standardised translation interface using a “dictionary”

The dictionary mechanism uses a syntax specification of CNC programming dialects in the form of XML to help the standardised interface to realise the translation from dialects to the Meta-model. The programming specification is a description of programming format and its correspondence representation or interpretation with the Meta-model. An XML description method is used to realise this objective. In this research it is called XML specifications of the programming dialects. All of the XML based NC programming specifications assembly a dialects-Meta dictionary, which works as a reference to assist the interface to translate CNC dialects into the Meta-model. Each XML specification is an entry of the dictionary. Hence, it is possible to translate any CNC dialect through the interface by simply adding a new entry to the dialects-Meta dictionary.

Using the dictionary method, a single standardised translation interface is needed between CNC part programme written in various dialects and the Meta-model, as shown in Figure 6.3. Consequently, the translation interface, universal process comprehension and the Meta-model can be encapsulated as a standalone system, which is given the name of UP*Ci*. The part programme and the XML specification of the dialect can be treated as the input to UP*Ci* and the output of UP*Ci* is STEP-NC representation of the process plan. This structure of UP*Ci* gives it necessary robustness and expansibility.

In Figure 6.5, the XML specification has been designed for Fanuc 18i controller. In this programming specification, the programming format for the controller is defined and mapped with the Meta-model. The root tag is the beginning of the XML file. The first part of the file is the detail about the specification to imply: which controller this schema is applied with, when the schema is created and by who. After that is the first part of the programming format: the overall grammar syntax about the ending, comments etc.

```

<Movements>
  <RapidPosition>
    <Modal>true</Modal>
    <Syntax>
      <Format>*$G0*$X@axis0$Y@axis1$Z@axis2</Format>
      <Parameters>
        <Optional>
          <Prm>$G00</Prm>
        </Optional>
        <Mandatory/>
      </Parameters>
    </Syntax>
  </RapidPosition>
  <LinearInterpolation>
    <Modal>true</Modal>
    <Syntax>
      <Format>*$G1*$X@axis0$Y@axis1$Z@axis2$F@feedrate</Format>
      <Parameters>
        <Optional>
          <Prm>$G1</Prm>
          <Prm>$F@feedrate</Prm>
        </Optional>
        <Mandatory/>
      </Parameters>
    </Syntax>
  </LinearInterpolation>
  <CIRCULARInterpolation>
    <Modal>true</Modal>
    <Syntax>
      <Format>*$G2*$X@axis0$Y@axis1$Z@axis2$R@radius$H@helical$F@feedrate</Format>
      <Parameters>
        <Optional>
          <Prm>$G2</Prm>
          <Prm>$R@radius</Prm>
          <Prm>$H@helical</Prm>
          <Prm>$F@feedrate</Prm>
        </Optional>
        <Mandatory/>
      </Parameters>
    </Syntax>
  </CIRCULARInterpolation>

```

Figure 6.5 - XML specification for Fanuc 18i controller

The second part of the schema is about the motion commands under the *Movement* tag. The first motion command in this file is the rapid position of the cutting tool. The value of the *Modal* tag is *true*, which indicates this command is continually in effect without being explicitly programmed in the next command line, as shown in Figure 6.6 (a). The line N180, N185 and N190 following the line N 175 mean the same linear-interpolation command and do not need to have “G1” explicitly programmed. The next tag is about the programming syntax for the command including the *Format*, *Parameters*. The *Format* tag defines the full presentation of the commands including all possible parameters. There can be more than one *Format* tags for a single command. For example, for Fanuc 18i controller, there are two format tags for clockwise circular interpolation to indicate there two options to programme the cutting tool to move in along a clockwise circular arc, as

shown in Table 6.1. The parameters of each command are defined in the *Format* tag and started with dollar (\$) symbols. The main command word(s) is embraced by two asterisks (*) on each side. In the *Parameters* tag, all of the parameters are listed with their functions. They are categorised into two groups: *Optional* and *Mandatory*, which indicate whether they are optional or compulsory for the command. For example in Figure 6.6 (b), the position words *X*, *Y* or *Z* are optional for a *G0* rapid position commands, while the *Z* indicating the drilling depth and the *R* indicating the radius of the hole are mandatory for the *G83* drilling cycle commands.

N175 G1 Z0.03 F1261.	N380 G0 G17 G54 X60.0 Y80.0 S1212 M3
N180 X54.57 Z-1.64	N385 G43 H3 Z25.0 M8
N185 X74.57 Z-3.31	N390 G83 G98 R3.0 Z-36.009 Q20.0F364.
N190 X54.57 Z-4.98	N395 G0 G80 Z25.0

(a) (b)

Figure 6.6 - Part programmes segments for Fanuc 18i controller

The mapping between the CNC dialects and the Meta-model is coupled with the help of NC programming specifications in the XML file. In the Meta-model, for instance, there is a Meta command of the linear interpolation ordering the cutting tool to cut along a straight line specified by the start and end coordinates. The Fanuc 18i uses *G1* for the linear movements, which is described in the corresponding XML file, as shown in Figure 6.5. The translation interface of UPCi reads in the XML file and stores the description in memory. Then it reads in part programme and interprets it line by line. The detailed process to translate the part programme to the Meta-model is illustrated in Figure 6.7.

As shown in Figure 6.7, the process to interpret a part programme starts from loading an appropriate XML specification. In the translation interface, there is an XML parser used to parse the XML file and store it in an XML parser object. Then the part programme is loaded in and handled line by line. For each line, a pre-process operation is applied to format the line into a standard presentation, and get rid of the comments etc. The XML parser object will provide necessary information regarding syntax grammars, such as comments indicators, line number words. With the standard format, the XML parser object will check the syntax of that line: how many command keys in that line, all of the mandatory keys included or not, find out the keys and put them into a linear array. Then traverse the array, match each key with the XML parser object to generate the

corresponding Meta-model object. For each Meta-model object, the translation interface will reference the XML parser object to acknowledge the parameter words and set values for the parameters of the meta-model object. After all of the command keys traversed, move to next line until the whole part programme processed the translation activity finishes. The part programme is represented by the Meta-model objects, which would be used as the input for process comprehension.

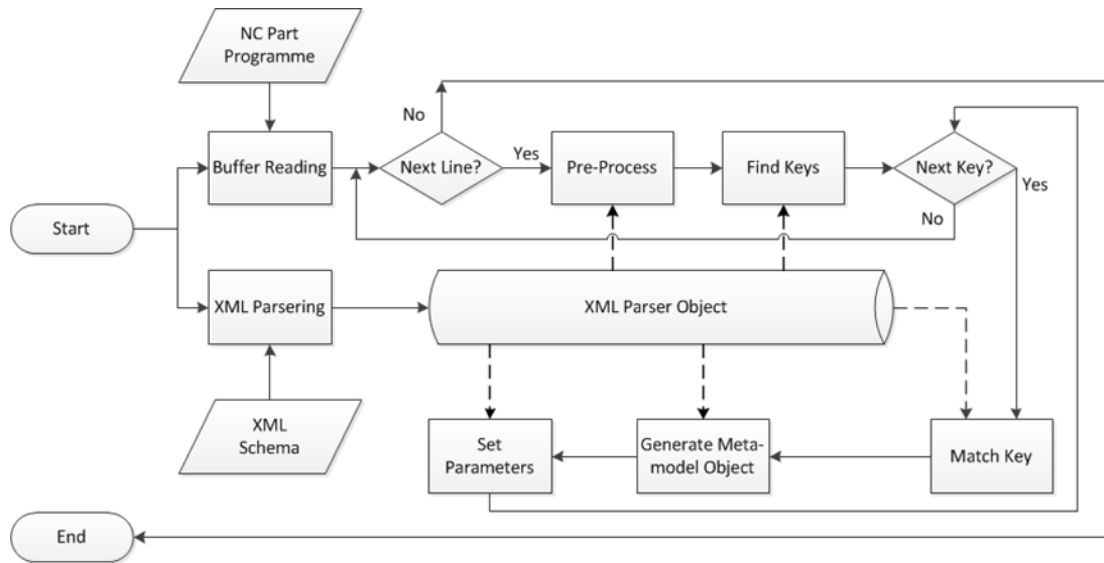


Figure 6.7 - Translation of part programme to Meta-model

The XML method used to describe the programming syntax enables UPCi to be expandable to support other programming languages. To achieve this, an XML file with the specification of the programming syntax is necessary. The users of UPCi can develop XML specifications according to their needs. In order to help the users to develop appropriate XML description files, a regulation is needed to make sure the XML files are in a uniform format to be suitable and understandable to UPCi. The regulation of XML file is used to define the legal building blocks of an XML specification of NC programming languages.

There are many XML regulation methods available, of which the most widely used are Document Type Definition (DTD) and XML Schema or XML Schema Definition (XSD) (Lee and Chu 2012). Compared with DTD, XSD is more powerful and widely used (Bex *et al.* 2004). Moreover, XSD has built in data types and allow the user to custom data types (W3C 2012), which is convenient to this research. Thus, XSD is chosen in this thesis to be used as the regulation method of XML specifications of NC programming languages. The XML Schema used to regulate the XML description of NC programming syntax has been

developed as shown in Figure 6.8. In this schema, the basic blocks used to describe the syntax of NC programming languages have been specified. Following the format, the user can develop XML specifications for other NC programming languages. In implementation, the XML specifications will be checked against this schema to validate the format of the specifications.



Figure 6.8 - XML Schema

6.5. Summary

This chapter explored the possibility to realise universal process comprehension of different CNC programming dialects. Based on the analysis and comparison of various CNC dialects, a Meta-model of CNC programming languages has been proposed. To translate CNC dialects into this Meta-model without developing loads of translation interfaces, a dictionary method is used. Programming specifications of CNC dialects have been modelled in XML format to realise the standardised translation of CNC dialects. The Meta-model together with the XML description of CNC dialects enables the UPCi to be a expansible system for new programming dialects. The XML schema is used to ensure the standardisation of the XML specifications of NC programming languages and entitle robustness to UPCi .

7. Feature recognition and knowledge capture from part programmes

7.1. Introduction

In the last chapter, the part programmes written in different CNC dialects have been translated to Meta-model objects. This is resource independent information of process plan data contained in the original part programmes. As discussed in Chapter 5, the following stage of process comprehension is to recognise the features and associated entities for a STEP-NC representation. In this Chapter, the feature recognition method will be discussed. Based on that, shopfloor knowledge capture and generation of STEP-NC representation are also addressed.

7.2. Feature recognition from part programmes for prismatic parts

Feature recognition in this research is different from traditional feature recognition as since the traditional feature recognition is based on the design data while in this research features are supposed to be derived from manufacturing data, more specifically part programmes. It is difficult to apply the methods (as reviewed in Chapter 4) published by other researchers directly. However, it does provide some helpful allusions to this research. The next section of this thesis will discuss the feature recognition methods employed in this research.

Before the discussion of feature recognition method, two points need to be clarified: feature recognition in this research is focusing on machining or manufacturing feature, since the part programmes are the shopfloor data about the manufacturing process of machining. Additionally, design features or features of other categories can be different from manufacturing feature. For instance, a boss is a normal feature in design. In manufacturing, it cannot be a feature by itself. It should be associated with other features such as pocket and planar face as an area inside of a feature, where the materials would not be machined. Another point is that feature recognition in this research is about prismatic parts. In other words the features involved in this research are all 2½D manufacturing features. For a prismatic part, there are two attributes needing to be identified through successful feature recognition: feature contour profile and feature depth.

7.2.1. Feature recognition from toolpath

Following the discussion in the last chapter, CNC part programmes can be represented by the Meta-model without concern of which programming dialects are used for the part programmes. Feature recognition in this section is started with the Meta-data interpretation of part programmes. This is basis of the feature recognition algorithms. Since feature recognition algorithms only deal with this Meta-data, a single set of algorithms is possible to handle part programmes written in different dialects. A general feature recognition process is proposed and illustrated in Figure 7.1.

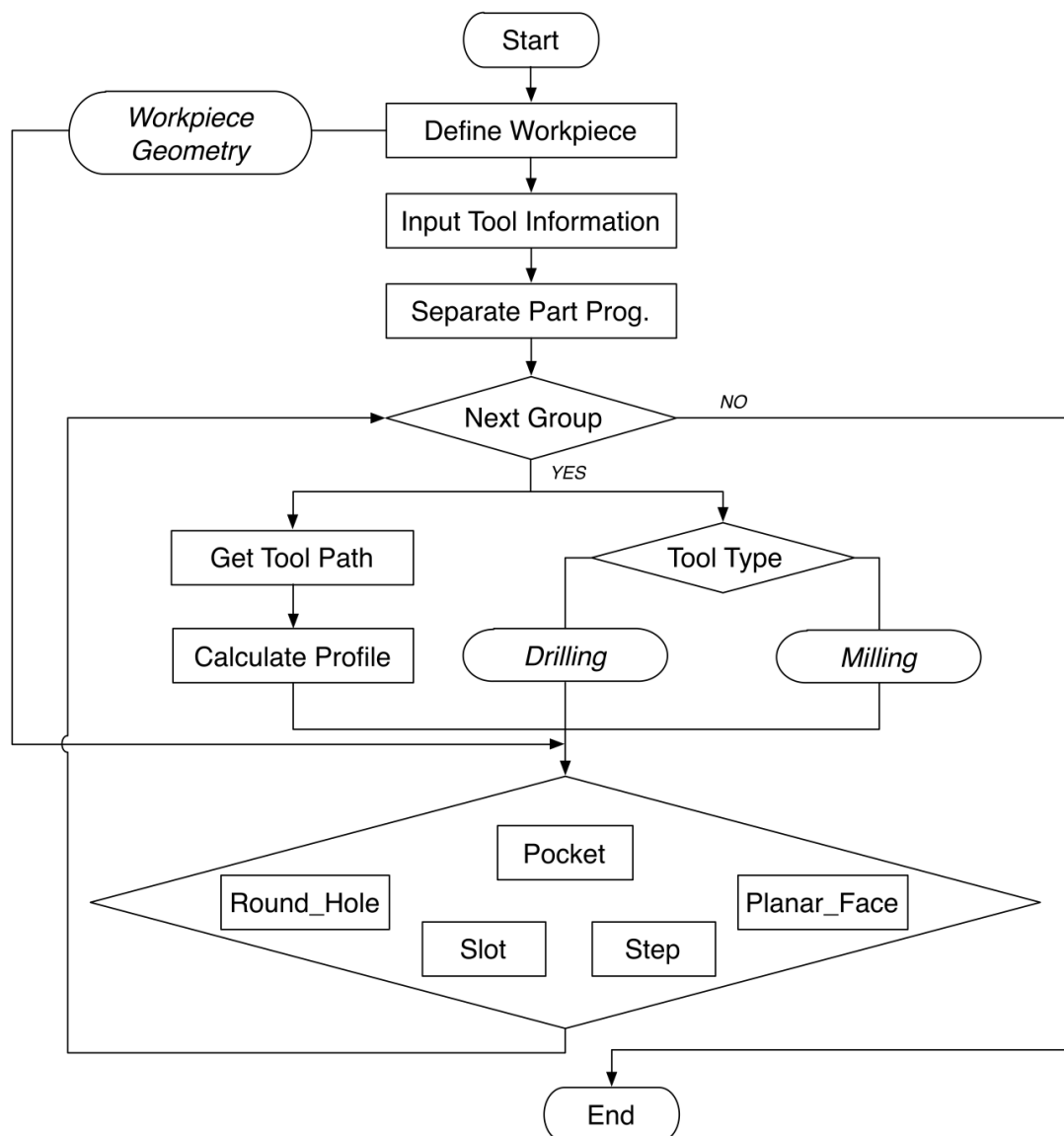


Figure 7.1 - Feature recognition process

As specified in Chapter 5, the input of the Universal Process Comprehension interface (UPCi) has three: part programme, workpiece geometry and cutting tool information. At this stage, part programmes have been translated into Meta-model objects. For successful feature recognition, workpiece geometry and cutting tool information are still needed to be specified. Hence the feature recognition process starts from defining workpiece geometry and cutting tool information. This information will then be used to identify the feature.

With all of the input information ready, the next stage is to divide the part programme (represented in Meta-data) into several sections. Each section is an operation associated with a machining feature. Then traverse all of the sections until the process ends. One feature should be identified from each section. The number of the features should equals to the number of operations. Since it is possible to have rough and finish operations for the same feature, the last stage of feature recognition process is to find out the operation associated with the same feature, and shortlist the final features. This final stage of the feature recognition is not included in Figure 7.1.

For the single feature recognition of each section of part programmes, the cutting tool information is vital. The cutting tool type can be used to identify the potential feature options. For example, a face milling tool can be used to machine a planar face, a step or even a slot. However it definitely cannot be used to machine a closed pocket. Drilling tools are mainly used to machine holes. After shortlisting the candidates, other information can be used to identify the feature. In each section of a part programme, traveling path of the cutting tool should be derived as shown in Figure 7.2 (a). The toolpath shown in the figure is from a roughing operation of a pocket. With the toolpath, the boundary of cutting area can be calculated from it. Offsetting the boundary of cutting area by the value of cutting tool radius, the feature boundary should be available. As shown in Figure 7.2 (b), the red rectangular is the boundary of the feature. Then compare the feature boundary with the workpiece boundary to identify the right feature. Through the process illustrated in Figure 7.1, the feature contour can be identified. Feature depth still needs to be identified. Since this research is applied with prismatic parts ($2\frac{1}{2}D$), it means there are no curved bottom surfaces in features. If there is any in the part programme and input into UPCI, it will ignore it. Hence, the depth of the feature can be simply identified through the deepest cut.

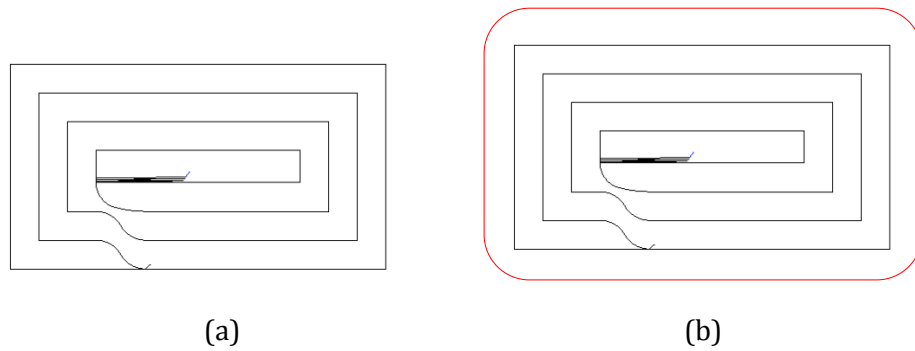


Figure 7.2 - Feature identification: cutting area of toolpath

The process illustrated in Figure 7.1 is a general feature recognition method used in this research. For each specific feature, this process can be different. In this following section, feature identification methods (the red rectangular part in Figure 7.1) used for each particular feature will be discussed in further detail. Several common prismatic features are employed to validate this feature recognition method.

7.2.1.1. Pocket

As discussed earlier, the toolpath boundary and cutting area can be calculated as shown in Figure 7.2. Comparing the cutting area with the workpiece boundary, the blue rectangular in Figure 7.3 (a): if the cutting area is entirely inside of the workpiece boundary, the feature should be a closed pocket, as shown in Figure 7.3 (b); Otherwise, it should be an open pocket (Figure 7.4).

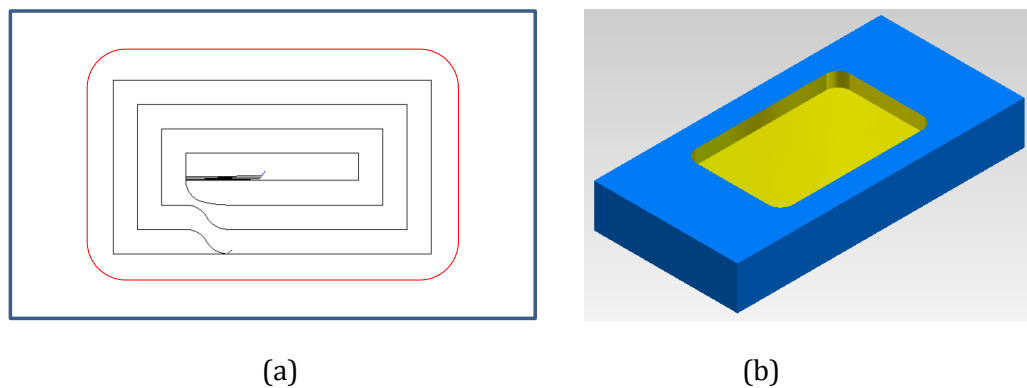


Figure 7.3 - Feature identification: a pocket

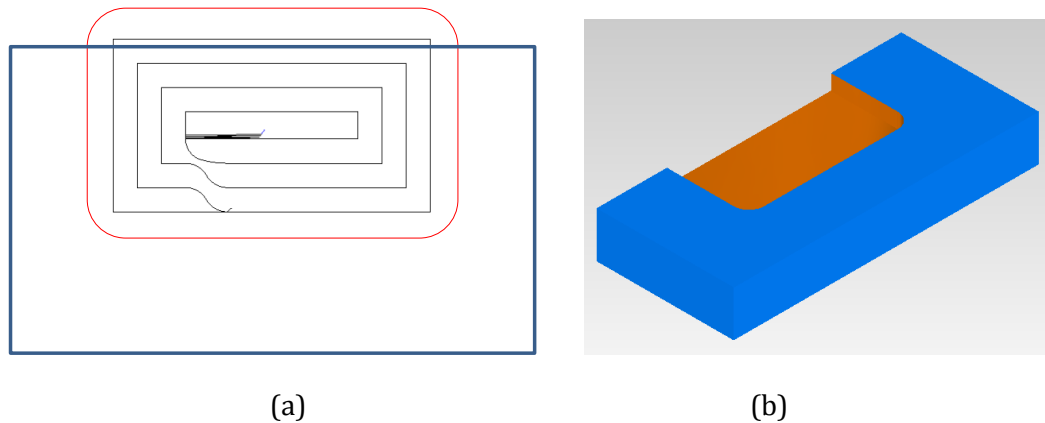


Figure 7.4 - Feature identification: an open pocket

In STEP-NC (ISO 14649-10 2002), the EXPRESS definition of a general pocket and its super types are:

```
ENTITY two5D_manufacturing_feature
ABSTRACT SUPERTYPE OF (ONEOF(machining_feature, replicate_feature,
compound_feature))
SUBTYPE OF (manufacturing_feature);
feature_placement:    axis2_placement_3d;
END_ENTITY;
```

```
ENTITY machining_feature
ABSTRACT SUPERTYPE OF (ONEOF(planar_face, pocket, slot, step, round_hole,
toolpath_feature, profile_feature, boss, spherical_cap, rounded_end, thread))
SUBTYPE OF (two5D_manufacturing_feature);
depth:                elementary_surface;
END_ENTITY;
```

```
ENTITY pocket
ABSTRACT SUPERTYPE OF (ONEOF(closed_pocket, open_pocket))
SUBTYPE OF (machining_feature);
its_boss:              SET [0:?] OF boss;
slope:                 OPTIONAL plane_angle_measure;
bottom_condition:      pocket_bottom_condition;
planar_radius:         OPTIONAL toleranced_length_measure;
orthogonal_radius:     OPTIONAL toleranced_length_measure;
END_ENTITY;
```

The pocket is derived from *machining_feature*, which is a subtype of *two5D_manufacturing_feature*. The other features discussed in next few sections are at the same level with pocket, subtypes of the *machining_feature*. There are two different pockets: closed pocket and open pocket. For each of them (ISO 14649-10 2002):

```
ENTITY closed_pocket
SUBTYPE OF (pocket);
feature_boundary:    closed_profile;
END_ENTITY;
```

```
ENTITY open_pocket
SUBTYPE OF (pocket);
open_boundary:        open_profile;
wall_boundary:        OPTIONAL open_profile;
(*)
Informal propositions:
- The entire open_boundary profile lies in the local xy plane.
- The open_profiles are not self-intersecting.
- Together the two open_profiles form a closed profile.
- wall_boundary is for information only.
*)
END_ENTITY;
```

If it is a closed pocket, the toolpath boundary is the feature boundary. For an open pocket, the open boundary of the feature should be the part of the toolpath boundary, the part lying inside the workpiece boundary. The wall boundary of an open pocket is optional. There is a need to assign it a value, which will be decided implicitly by the selected tool and the fillet options (ISO 14649-10 2002). The *feature_boundary* and *open_boundary* are the description of the upper edge of the pocket. From the definition of a general pocket, another attribute for a pocket is their bottom condition. There are several options defined in ISO 14649-10 (2002):

```
ENTITY pocket_bottom_condition
ABSTRACT SUPERTYPE OF
(ONEOF (through_pocket_bottom_condition, planar_pocket_bottom_condition,
radiused_pocket_bottom_condition, general_pocket_bottom_condition));
END_ENTITY;
```

In the case of this research, there two options: through and planar (not through) for prismatic parts. Since the setup and workpiece information are available, the toolpath can be used to judge is the pocket through or not. If the deepest cut is exceeding the workpiece boundary, it is a through pocket. Otherwise it is a pocket with a planar bottom.

Another issue with pocket is how to differentiate a pocket from other features such as slot or round hole. Since a slot or a round hole can be considered as a special kind of pocket. In STEP-NC (ISO 14649-10 2002), slot is listed a separate feature in order to compatible with ISO 10303-224 (ISO 10303-224 2000), which is a part of STEP to describe mechanical parts using machining features. In this research, the situation illustrated in Figure 7.4 is treated as an open pocket. However, the case is different if the cutting tool diameter equals the open pocket length. In the case, the feature, machine by a single sweep of the tool, is recognised as a slot. Similarly, if a feature is machined with a straight line of toolpath, it is recognised as a round hole with the same diameter of the tool. Otherwise, if the toolpath is a closed circle, the feature is recognised as a round pocket. This specification does not include the situation that a drilling cycle is used, in which case the feature is definitely a round hole. The round hole recognition method is discussed in Section 7.2.1.6.

7.2.1.2. Slot

If the case is like the situation shown in Figure 7.5 (a), the cutting area is covering part of the workpiece with two ends open. The feature should be a slot as shown in Figure 7.5 (b). The only difference from an open pocket is that it has two open ends. As discussed in the previous section, it should be recognised as a slot if the feature is manufactured by a single sweep of the tool. Hence, the case shown in Figure 7.6 (a) is recognised as a slot as well (Figure 7.6 (b)).

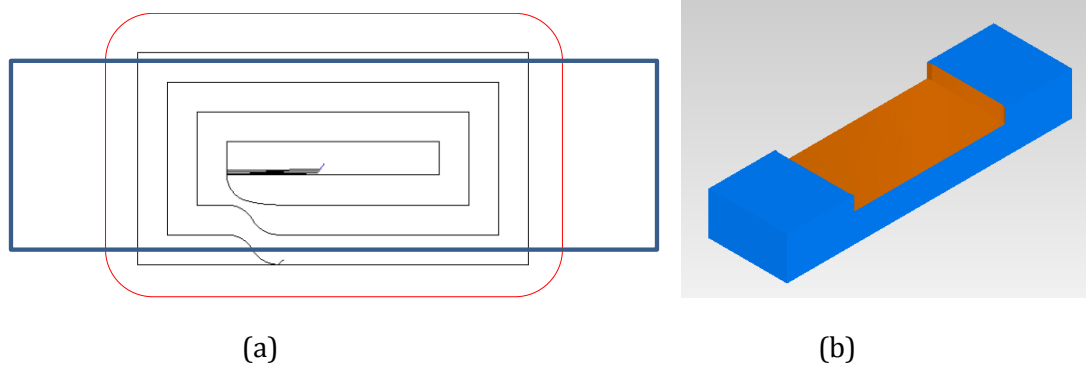


Figure 7.5 - Feature identification: a slot

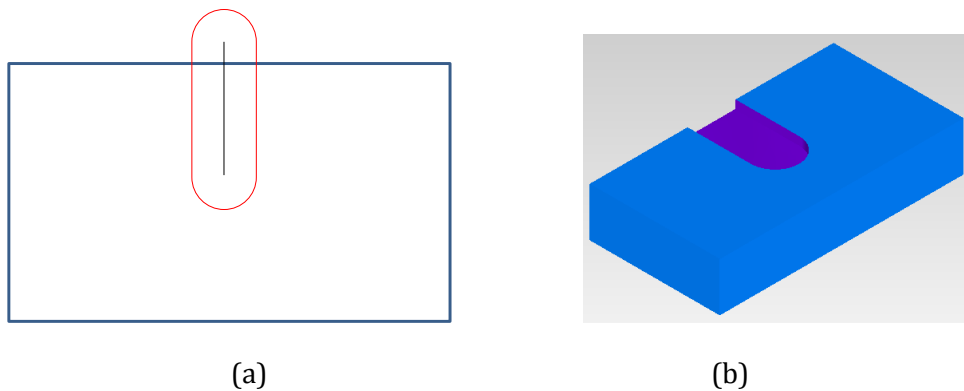


Figure 7.6 - Feature identification: a slot

In STEP-NC, the EXPRESS definition of a slot is (ISO 14649-10 2002):

```

ENTITY slot
  SUBTYPE OF (machining_feature);
  course_of_travel:    travel_path;
  swept_shape:         open_profile;
  end_conditions:      LIST[0:2] OF slot_end_type;
END_ENTITY;
```

The *course_of_travel* in the definition here is tool path extracted from the part programme, along which the tool is traveling. As shown in Figure 7.6 (a), the *course_of_travel* is the straight line in the middle of the slot. The *swept_shape* is the cross-section generated by the tool, which only used to specify the width of the slot. The *end_conditions* of a slot have several options (ISO 14649-10 2002):

```
ENTITY slot_end_type
```

```
ABSTRACT SUPERTYPE OF (ONEOF (woodruff_slot_end_type,  
radiused_slot_end_type, flat_slot_end_type, loop_slot_end_type,  
open_slot_end_type));
```

```
END_ENTITY;
```

There are three options for prismatic parts: *radiused_slot_end_type*, *flat_slot_end_type* and *open_slot_end_type*. However, according to author's clarification between a slot and a pocket, there is no chance to have a slot with a *flat_slot_end_type* end. Hence, if an end of the travel path of the tool for a slot is inside of the boundary of workpiece, it is a *radiused_slot_end_type* end. Otherwise, it is a *open_slot_end_type* end. It is worth to point out that all of discussion above about the attributes of a slot is based on the assumption that the slot was machined by the single sweep of the tool.

If the case is like the situation illustrated in Figure 7.5 (a), the slot is machined by a tool with a smaller diameter than its width by multi traveling. The attribute *course_of_travel* of the slot should not be the toolpath anymore. Instead, it should be the centre line of the slot with the same length. However, it is just the manner of STEP-NC to acknowledge a slot. It does not necessarily mean that the slot should be machined by a tool with the diameter equalling to its width. In machining, it is more likely to use a small tool to perform more than one cut to machine the feature.

7.2.1.3. Step

If the case is like the situation shown in Figure 7.7 (a), the cutting area is covering part of the workpiece at one end. The feature should be a step as shown in Figure 7.7 (b).

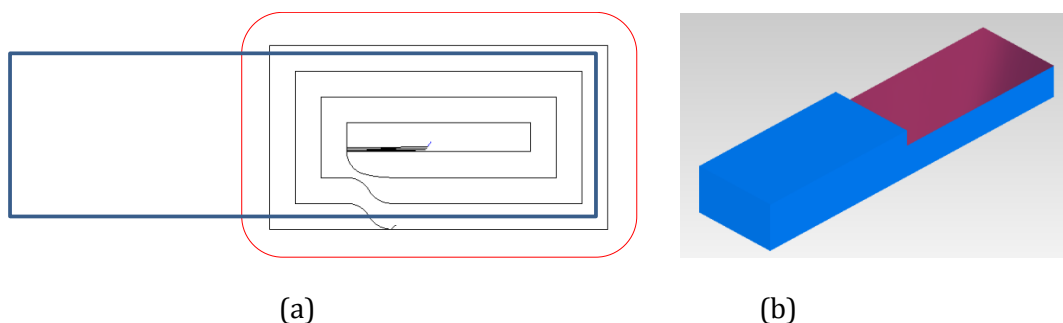


Figure 7.7 - Feature identification: a step

```
ENTITY step
```

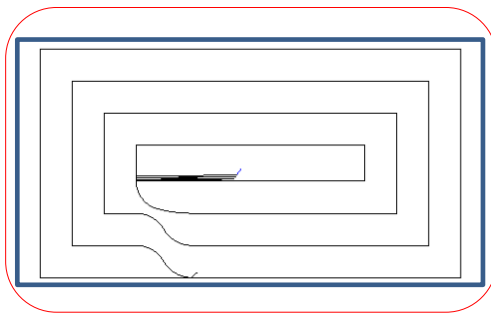
```
SUBTYPE OF (machining_feature);
```

```
open_boundary:          linear_path;  
wall_boundary:          OPTIONAL vee_profile;  
its_boss:               SET[0:?] OF boss;  
END_ENTITY;
```

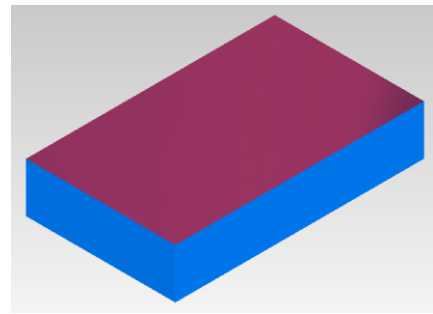
The *open_boundary* of the step is the outline that forms the upper edge of the feature. The outline or shape that forms the side edge of the step is the *wall_boundary* of the feature.

7.2.1.4. Planar face

If the case is like the situation shown in Figure 7.8 (a), the cutting area is covering the top face of the whole workpiece. The feature should be a planar face as shown in Figure 7.8 (b).



(a)



(b)

Figure 7.8 - Feature identification: a planar face

In STEP-NC, the definition of a planar face is (ISO 14649-10 2002):

```
ENTITY planar_face  
SUBTYPE OF (machining_feature);  
course_of_travel:      linear_path;  
removal_boundary:      linear_profile;  
face_boundary:         OPTIONAL closed_profile;  
its_boss:              SET [0:?] OF boss;  
END_ENTITY;
```

The *course_of_travel* and *removal_boundary* are straight lines with magnitude and direction specified. The boundary of the planar face is defined by sweeping the *removal_boundary* along the *course_of_travel*. The *face_boundary* is the final shape of the workpiece after the planar cut has been applied. The definition is illustrated in Figure 7.9.

Since the covering area of the toolpath has been calculated, there can be many options for the *course_of_travel* and *removal_boundary* of the feature to cover the whole area. In the case shown in Figure 7.8, the *course_of_travel* and *removal_boundary* can be simply the two adjacent edges of the rectangular.

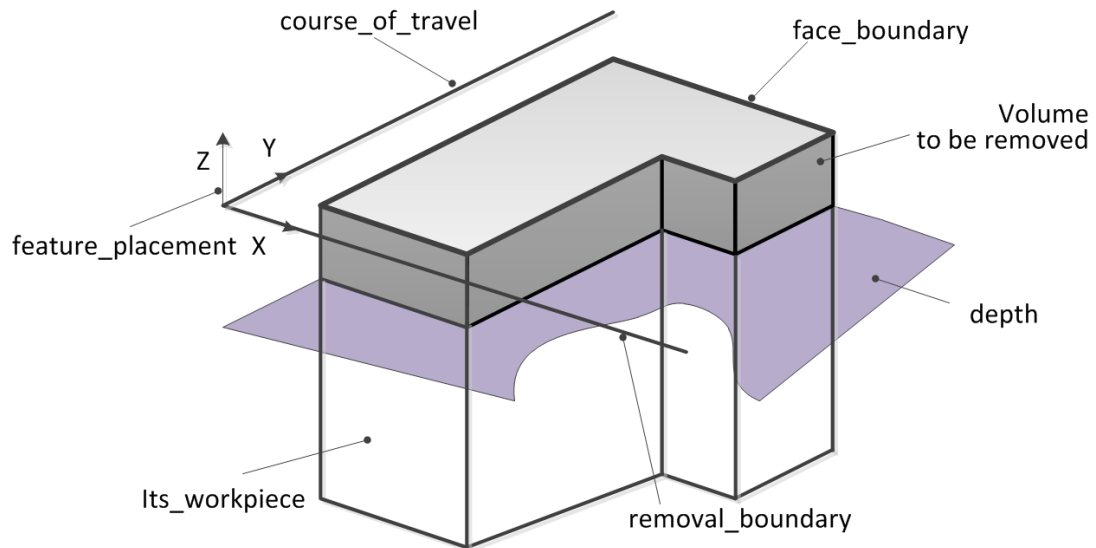


Figure 7.9 - Feature definition in STEP-NC (ISO 14649-10 2002)

7.2.1.5. Boss

A boss is that material which remains unworked after the machining of a feature with a boss. For this reason, boss is not an independent machining feature but exists only as an attribute of others (ISO 14649-10 2002). In fact, from the definitions of previous features in this section including planar face, pocket, step, they may have one or more bosses. From this point of view, a feature with one or more bosses is a combination of two or more features. This will lead to an issue in traditional feature recognition: interacting features. This topic is discussed later in the discussion chapter of this thesis. Since this research is based on prismatic parts, a boss with a slope as shown in Figure 7.10 (a) is not considered. The boss type without the slope surface as shown in Figure 7.10 (b) is the one this research deals with. The STEP-NC definition of a boss is (ISO 14649-10 2002):

```
ENTITY boss
SUBTYPE OF(machining_feature);
its_boundary: closed_profile;
slope:      OPTIONAL plane_angle_measure;
END_ENTITY;
```

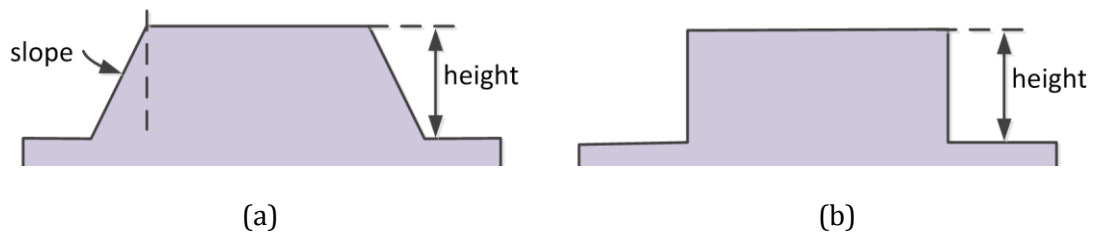


Figure 7.10 - Two types bosses

For the recognition of a boss, it is also connected with the recognition process of the mother feature where the boss locates. Henceforth, the term of mother feature will be used to refer the feature to which a boss is attached. From the definition of a boss, it does not have a height attribute. Since a boss is the remaining area of another feature, the depth of a boss should be the same with the depth of the mother feature. The boundary of the mother feature should be different from the same feature without bosses. It should have one or more inside boundary to indicate where the material should remain. These inside boundaries are the shape of the bosses. Hence, when the feature boundaries of the mother features have been identified, it is simple to recognise the bosses inside them.

7.2.1.6. Round holes

A round hole is another common feature in prismatic parts. In CNC programmes, there are two methods to programme the CNC machine: using built-in drilling cycles of the programming languages or specify the detailed toolpath. The drilling cycles are discussed in Section 7.2.2. This section focuses on the toolpath method. From the view of technology, basically there are two types operations to create a hole: drilling or milling. For the first situation, a round hole is different from other features discussed in previous sections since a drilling type tool is needed to create a hole in the workpiece. It is easy to tell that the feature is a round hole from the tool type and the diameter of the hole equals to the diameter of the tool. For the second situation, if the toolpath is a straight line cutting into the workpiece, it is round hole. Otherwise, as discussed in Section 7.2.1.1, when the toolpath boundary is a full circle, it is treated as a round pocket. The definition of a round hole in STEP-NC is:

```
ENTITY round_hole
  SUBTYPE OF (machining_feature);
  diameter:          toleranced_length_measure;
```

```
change_in_diameter:  OPTIONAL taper_select;  
bottom_condition:    hole_bottom_condition;  
END_ENTITY;
```

The reference point of the location of the hole is the centre point of the top circle. In this research the tapered hole is not in consideration. The bottom condition has two options:

```
ENTITY hole_bottom_condition  
ABSTRACT SUPERTYPE OF (ONEOF (blind_bottom_condition,  
through_bottom_condition));  
END_ENTITY;
```

```
ENTITY blind_bottom_condition  
ABSTRACT SUPERTYPE OF (ONEOF(flat_hole_bottom,  
flat_with_radius_hole_bottom,  
spherical_hole_bottom, conical_hole_bottom))  
SUBTYPE OF (hole_bottom_condition);  
END_ENTITY;
```

The identification method of bottom condition is quite similar with the way how a pocket bottom condition is recognised: comparing the deepest cut with the workpiece boundary to decide whether it is a blind or through hole. For the blind hole, two types of bottom condition are considered: *flat_hole_bottom*, *conical_hole_bottom*, as shown in Figure 7.11. The conical hole condition is only formed by the shape of the drilling tool, which is an exception of the definition of prismatic parts, as mentioned earlier. Hence, the hole bottom condition is depending on the shape of tool head.

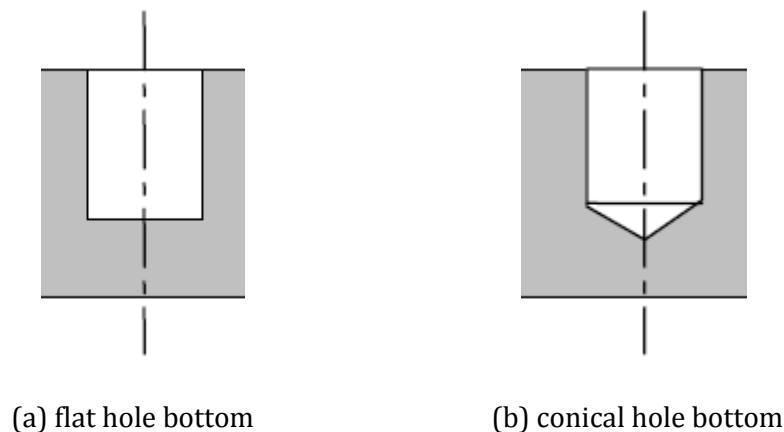


Figure 7.11 - Hole bottom conditions

7.2.2. Feature recognition from canned cycles

The canned cycles in CNC programming languages are pre-programmed subroutines that obviate the complex programming of a series of machine tool movement to machine a specific type of feature. In CNC programming a simple command is called with necessary parameters and the CNC machine does the rest. Typically there are canned cycles for drilling operations although there are milling cycles. For example Siemens has milling cycles: face milling CYCLE 71 and contour milling CYCLE 72. The milling cycles are rare and here only drilling cycles are covered. In a drilling operation, a rotating cutter makes a round hole into a stationary workpiece to a certain depth, as shown in Figure 7.12. Drilling cycles may include other parameters to define different drilling strategies to create holes for different purposes. In ISO 6983, several drilling cycles have been defined to create different holes. These canned drilling cycles are the focus of this section. Basically, from these drilling cycles to recognise the feature is quite straightforward, since it is clear that the feature is a round hole. To identify the feature, the diameter and the depth are needed. Apparently, the diameter of the hole equals to the diameter of the drilling tool. The depth of the hole can be found in the drilling cycle's parameters.

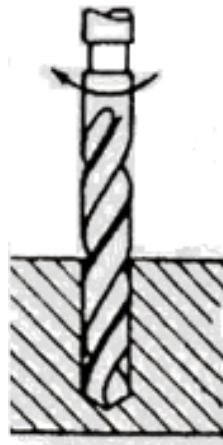


Figure 7.12 - Drilling operation

To a certain extent, the drilling cycles are feature based operations. They are different from the toolpath based part programmes, in which case the machine has no idea of the aim of cutting movements. In drilling cycles, there is no toolpath defined, the CNC machine decides the cutting toolpath by itself according to the cutting parameters defined in the cycles. Hence, the feature recognition process for the canned cycles is different from the approach discussed in previous Section 7.2.1. In Section 7.2.1, the features are identified and then associated process information (operations) could be generated according to the feature (in Section 7.3). For drilling cycles, it is possible to generate STEP-NC entities of operations for the features (holes) and then identify the features' geometry information (depth) according to the operation parameters. The method to generate operations from drilling cycles and identify the features geometry is presented in Section 7.3.

7.2.3. Feature identification

Feature identification issue is a unique one of feature recognition from part programmes since there is no such problem in traditional feature recognition. This issue arises from the fact that there is usually more than one operation applied on the same feature. Most frequently, there are two: roughing and finishing. Since there is a machining allowance for the finish operation, the features recognised from the rough operation and finish operation are different. For example, in Figure 7.13 there are operations for a planar face with a boss. The green part is the covering area of the toolpath. It is worthy to note that the covering area could be bigger than the actual cutting area. In Figure 7.13 (a), the rough operation is cutting out a boss at the top surface with allowance at the wall of the boss. In Figure 7.13 (b), the finishing operation finishes the wall of the boss. Hence, in

feature recognition the first operation can be recognised as planar face with a boss. The second operation can be recognised as a pocket with a smaller boss. The difference between the two bosses is the finish allowance. However, the original process of the part programme is a planar face with a boss needing two operations. This is the identification issue of feature recognition from the part programmes.

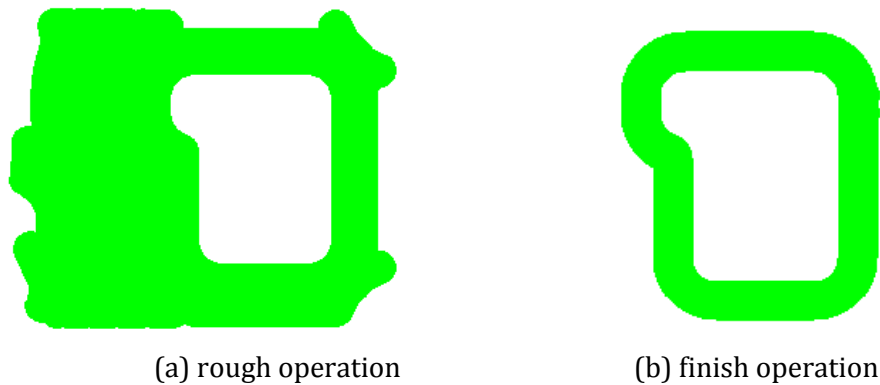


Figure 7.13 - Cutting area of operations of a boss

For the example in Figure 7.13, it is possible to treat the two operations as two features. In that case, the final part is the same after the two features have been machined. However, it is not good practice since it is different from the original process. Moreover, if the generated process plans in STEP-NC representation is used on another CNC machine, a pocket should be exactly the same covering area of the finish operation. To machine the pocket, the covering area of the toolpath for the pocket should be exactly the same as well, which is meaningless. Since the covering area is not the actual cutting area, normally, it is overlapping with the previous operation. In this case, a huge of effort would have been carried out to make the pocket which does not actually exists in the final part. Hence, it is necessary to keep STEP-NC representations of process plans consistent with the part programmes. Consequently, there is a further need to identify and remove redundant features.

However, it is challenging to identify and remove redundant features since they have different geometries. The example in Figure 7.13 is a simple one. The two bosses are similar in shape and the boundaries are offsetting a small distance. It is possible to identify the feature automatically. However, in some cases the number of the operations is not limited to two. For example in Figure 7.14, there are two rough operations and one finish operation. The second rough operation is used to shape the corner of the pocket.

The final operation is finishing the wall of the pocket. From the figure, the covering area of second rough operation is totally different with the shape of the pocket. Hence, it is difficult to judge whether the operation is applied to the pocket or not. In the author's opinion, it is necessary to use human intervention to help the recognition process. For those situations where the algorithm cannot handle, an optional choice can be promoted to the operator to decide. Since the toolpath can be visually available, it is easier for the operator to make a judgement.

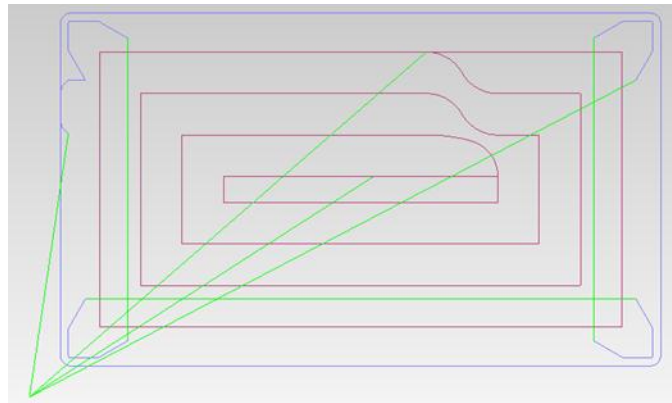


Figure 7.14 - Operations for a pocket

7.3. Knowledge capture from part programmes

As discussed in Chapter 4, the shopfloor is a knowledge intensive stage of manufacturing. A sophisticated part programme is the result of an operator's attempt and experiments. There is tacit knowledge of them behind the part programmes. Reusing machining process knowledge offers the industry opportunities to improve manufacturing traceability, quality, and control while enabling savings in cost and time. Rapid product development relies heavily on quick and reliable process planning and knowledge reuse to facilitate the generation of process plans efficiently and effectively.

At the shopfloor, manufacturing knowledge includes the machining sequence of a set of features, tool choice, machining strategies, parameter selection according to machining conditions, use of machine functions etc. Nearly all of the manufacturing knowledge except setup information is included in part programmes. On the other hand, STEP-NC does provide accommodation mechanism for all of the manufacturing knowledge, as shown in Figure 3.2. In this figure, the workingsteps are the sequence of the operation carried out. *Cutting_tool* entity is used to describe the cutting tool, which is one of the inputs of UPCi. All other manufacturing knowledge is included in the technology area,

and in fact, is covered by the machining operation. The detailed definition of machining operation is shown in Figure 7.15 and Figure 7.20. The inheritance relationship between different operation types in STEP-NC is illustrated. At the root of tree, the abstract entity *Machining_operation* has several attributes including cutting tool, machine function and machining technology. In Figure 7.16, the definitions of machine function and technology are presented. Along the heritance chain, other specifications concerning the machining method are added. The non-abstract operations at the end of the tree are the ones used in STEP-NC presentation files. As shown in these figures, with the accumulation along the heritance chain, the non-abstract entities at the end of the tree provide encapsulations of the manufacturing knowledge discussed here. Hence, the procedure to create proper operation entities based on part programmes is the process to restore the manufacturing knowledge from the shopfloor. The order of these operations is the machining sequence. This section mainly focuses the method to generate appropriate STEP-NC operation entities. Due to differences between milling type operations and drilling type operations, there are two sections to cover these two types of operations.

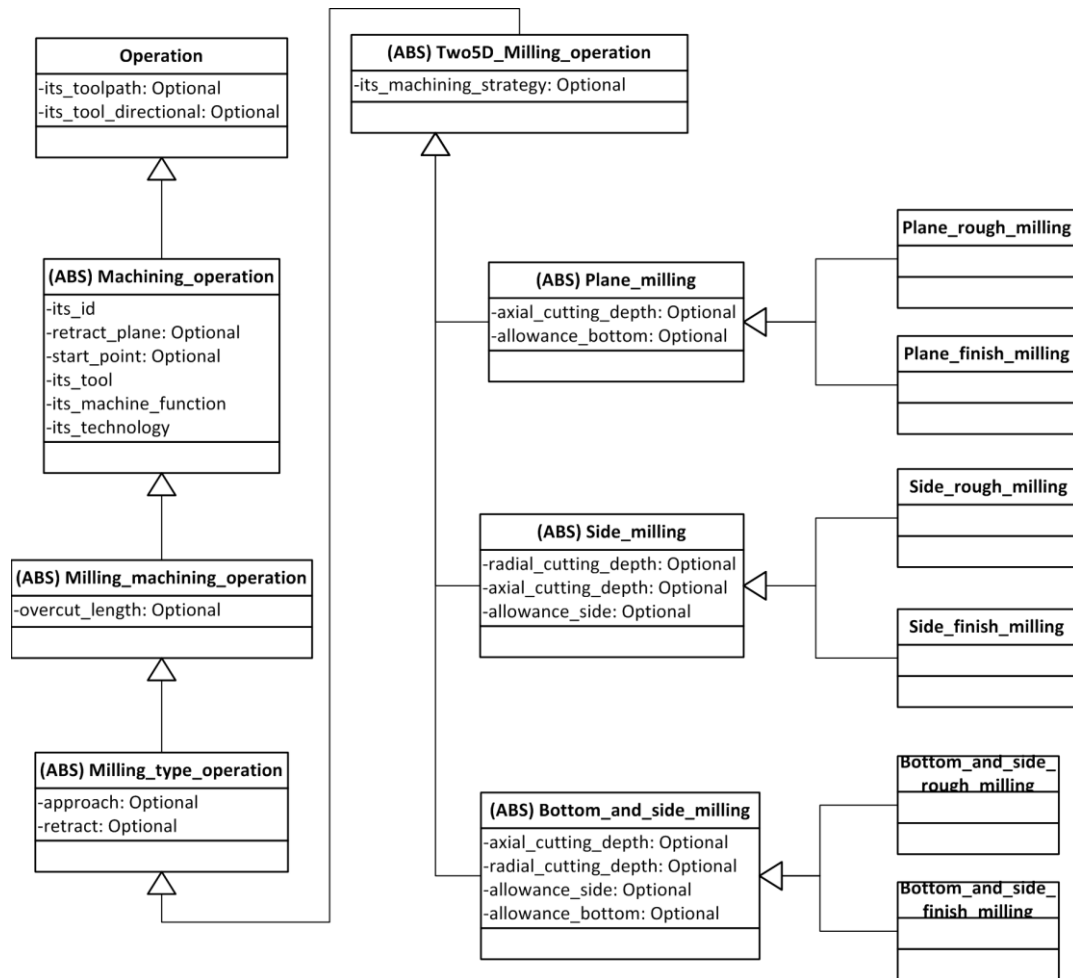


Figure 7.15 - Milling type operations in STEP-NC (ISO 14649-11 2002)

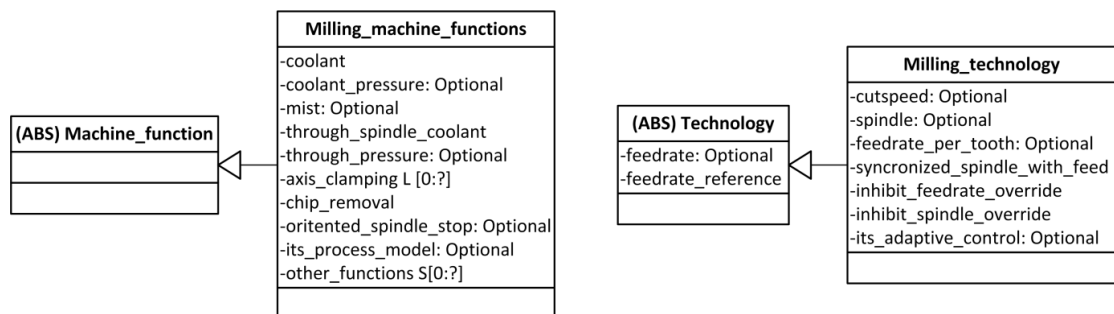


Figure 7.16 - Machine function and technology in STEP-NC (ISO 14649-11 2002)

7.3.1. Milling type operations

This section manages to generate operation entities from toolpath based part programmes. Most of them are milling type of operations. The generation of drilling type operations is discussed in the next Section 7.3.2. Basically, to create STEP-NC entities of operation, there are five main items to identify: operation types, cutting tool, machine

functions, technology and machining strategy. Since the cutting tool information is the one of the inputs of UPCi, there are four left. The situation applies with both milling and drilling type of operations.

The most important and straightforward task is to identify the operation types. In Section 7.2, the manufacturing feature recognition method has been discussed. Since the feature is identified, it is simple to identify the right operation type from three options listed in Figure 7.15: plane milling, side milling and bottom-side milling. For the features in Section 7.2, the operation of a planar face is the type of plane milling and the others are bottom-side milling operations. The last operation for a feature is the finish operation. If there is more than one operation applied on the same feature, the others are roughing operations. In fact, the difference between roughing and finishing is not technically strict or absolute. It can be noted from that the STEP-NC definitions of roughing and finishing operations are the same machining technology. For example in Figure 7.15, for plane milling, there are roughing and finishing operations, which share the same set of attributes. The other attributes like depth and allowance can be calculated with comparison with the feature geometry.

The machine function and technology can be derived from part programmes. It is quite straightforward to acquire the information from part programmes and set to STEP-NC entities. Table 7.1 shows the mapping between Siemens commands with STEP-NC entities. STEP-NC provides corresponding accommodations for these machine functions and technology parameters, while not limited to them. For example, in the Siemens part programme or even other dialects based on G&M codes, there are no limitations on overriding feedrate and spindle speed. In CNC machining, the operators can rotate override wheels on CNC machines to adjust the spindle speed or feedrate.

The last task is to identify the strategies used in each operation, including approach/retract strategies and machining strategies. From the EXPRESS-G diagram of different approach and retract strategies in Figure 7.17, there are six types approach and retract strategies. In Figure 7.18, there are eight types two5D milling strategies. In each group, there is a toolpath based strategy, *along_path* and *Explicit_strategy*, to specify complex strategies. Technically, it is possible for STEP-NC to include all strategies used in milling operations. Since there are so many different types of strategies, it is possible but difficult to develop algorithms to recognise each of them precisely, especially for some

similar ones such as *Bidirectional_contour* and *Contour_bidirectional*. Hence, in this research a human intervention method is used to identify the strategies, although some attributes are derived from the toolpath automatically such as the attribute overlap. The value of the overlap attribute is a percentage of cutting tool diameter to indicate the overlap between adjacent cutting movements, which can be easily figured out by the tool diameter and the distance between two adjacent cuts.

Table 7.1 - Machine functions and technology mapping between Siemens and STEP-NC

<i>Machine functions & Technology</i>	<i>SIEMENS commands</i>	<i>STEP-NC Entities</i>
Main spindle on clockwise	M3	spindle
Main spindle on counterclockwise	M4	
Main spindle off	M5	
Spindle speed	S	
Coolant on/off	M8/M9	coolant
Feedrate	F	feedrate
Constant cutting speed on/off	G96/G97	synchronised_spindle_with_feed
Feedrate override	N/A	inhibit_feedrate_override
Spindle override	N/A	inhibit_spindle_override

Using the method discussed in Chapter 6, the toolpath information can be extracted from the part programmes. It is possible and easy to visualise the toolpath to help the user to identify the strategies. For example, in Figure 7.19 there are two sets of toolpath for a planar face and a pocket. When they are presented to a user, it is easy to recognise the strategy. In Figure 7.19 (a), the milling strategy is a *Bidirectional*. The machining strategy in (b) is a *contour_parallel*. The approach and retract strategy for the pocket is a *plunge_zigzag*, as shown in Figure 7.19 (c). The attributes of the strategies can be entered by the user as well.

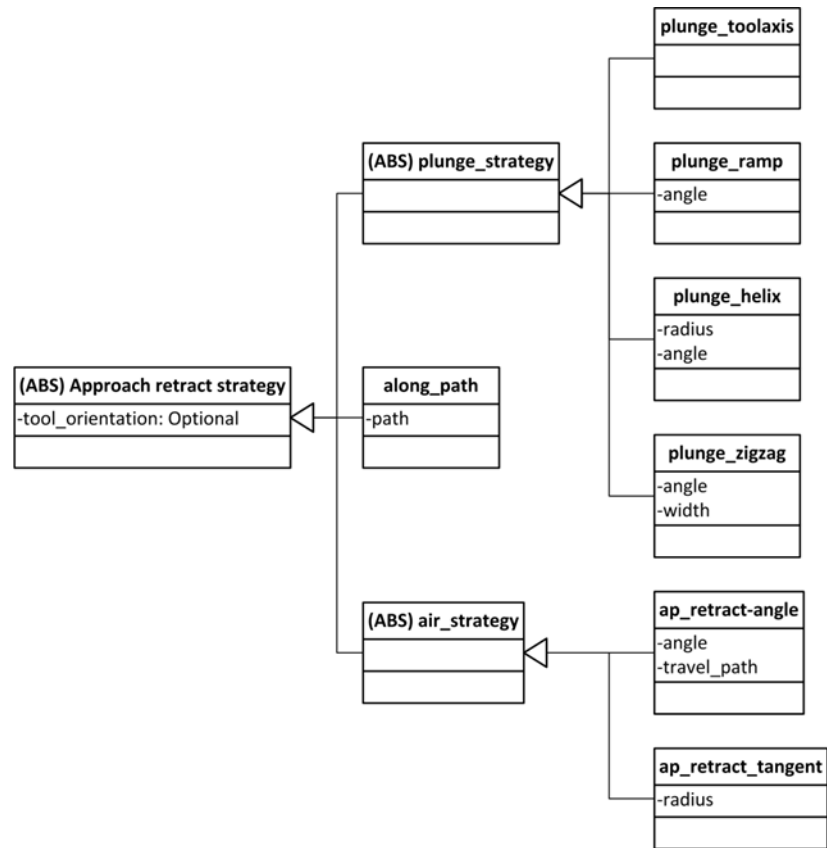


Figure 7.17 - Approach and retract strategies in STEP-NC (ISO 14649-11 2002)

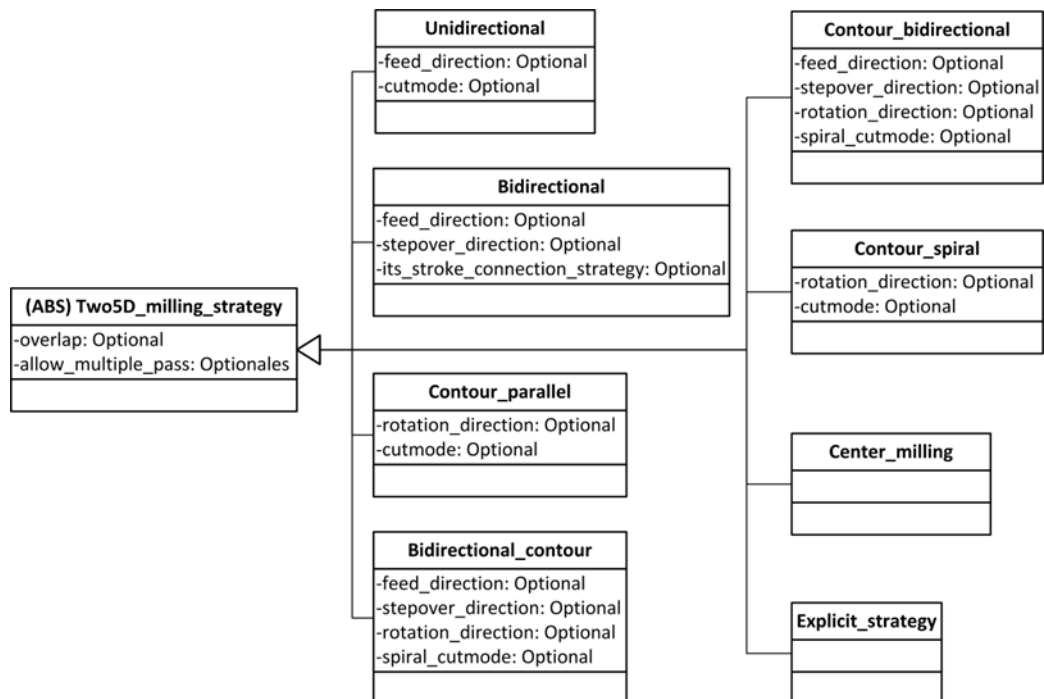
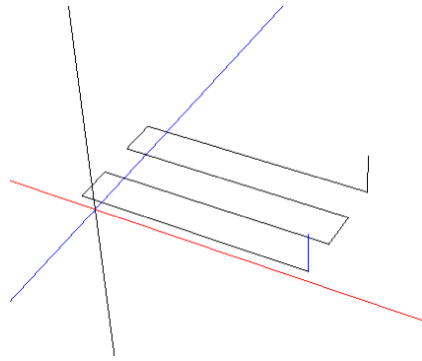
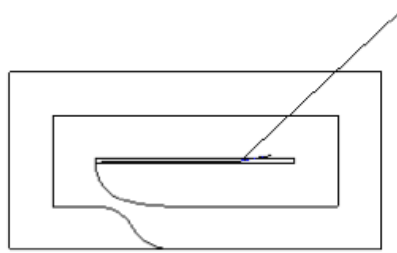


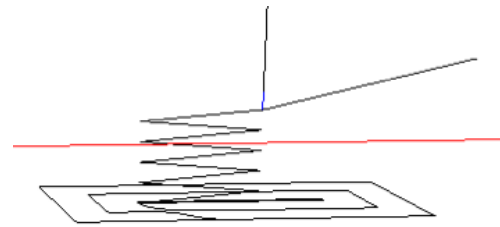
Figure 7.18 - Machining strategies in STEP-NC (ISO 14649-11 2002)



(a) toolpath of planar facing



(b) toolpath of a pocket (top view)



(c) toolpath of a pocket (side view)

Figure 7.19 - Toolpath visualisation for strategy identification

7.3.2. Drilling type operations

The drilling type operation is used to machine a round hole or cut a thread. The thread feature is out of the scope of this research. Hence, the drilling operation is only for round holes. As discussed in Section 7.2.1.6, in CNC part programmes, the drilling operation are programmed in two different ways: toolpath specification or canned cycles. For the first situation, the drilling type operation only applies when the toolpath is a straight line cutting into the workpiece as specified in Section 7.2.1.1 and Section 7.2.1.6, where the feature recognition method has been presented as well. In practices, there are few occasions where toolpath based drilling operations are used since it is too complex to programme. Hence, in this section, the recognition of the drilling operations is mainly focusing on canned drilling cycles.

In Table 7.2 (Siemens AG 2000; GE Fanuc Automation 1998), the mapping between STEP-NC and canned cycles from Fanuc and Siemens is listed. Basically, there is a corresponding entity for each canned cycle from Fanuc and Siemens. The drilling

operations in STEP-NC are shown in Figure 7.20. Since boring and reaming is technically similar except their cutting tools, and some occasions they are used interchangeably, boring and reaming are listed together in this table. As mentioned in Section 7.2.2, drilling cycles have feature based characteristics. The generation method of STEP-NC operation entities is different from milling type operations. Actually, the STEP-NC entities are quite identical with the canned cycles in terms of their attributes or parameters.

Table 7.2 - Mapping between STEP-NC entities and canned cycles of Siemens and Fanuc

STEP-NC entities	Siemens cycles	Fanuc cycles
Drilling	CYCLE 81	G81
Center_drilling	CYCLE 81	G81
Counter_sinking	CYCLE 82	G82
Multistep_drilling	CYCLE 83	G83
Back_boring	N/A	G87
Tapping	CYCLE 84 – Rigid tapping CYCLE 840 – Tapping with compensation chuck	G84
Boring/Reaming	CYCLE 85/86/87/88/89	G85/86/87/88/89

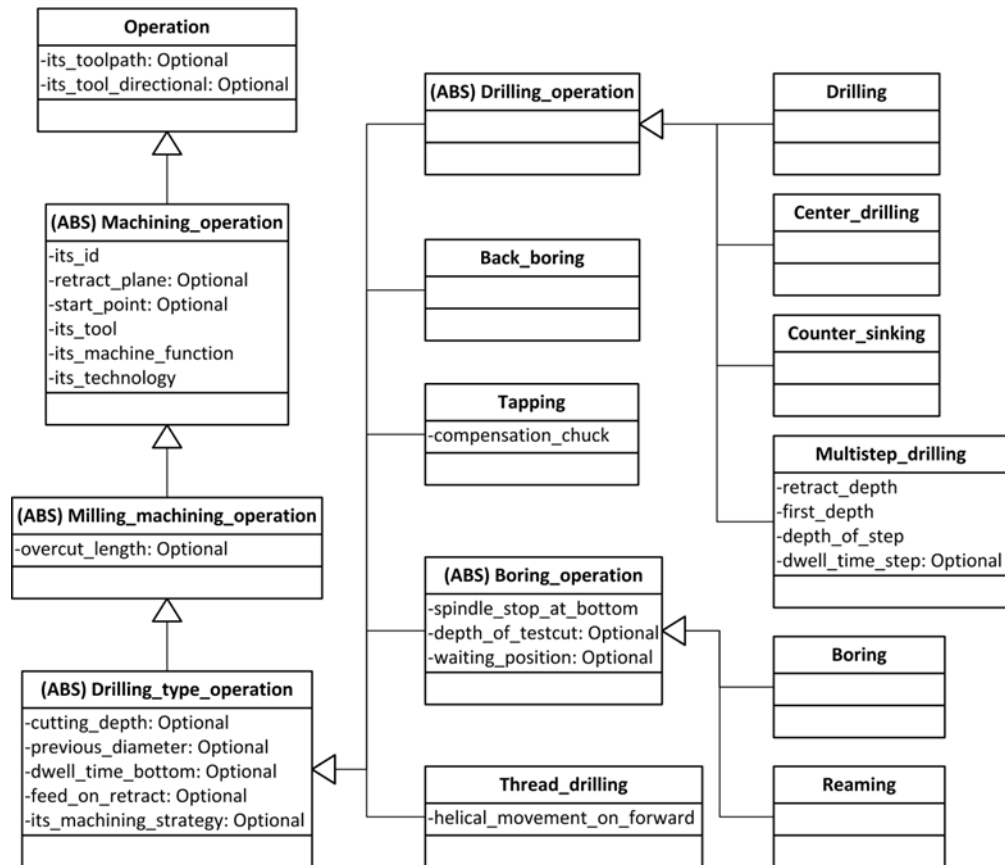


Figure 7.20 - Drilling type operations in STEP-NC (ISO 14649-11 2002)

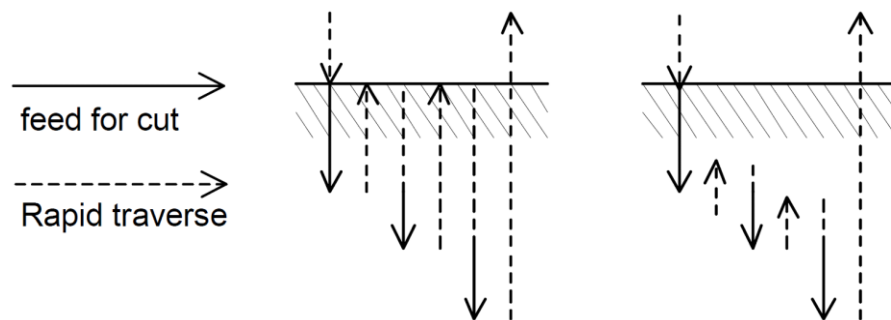


Figure 7.21 - Multistep drilling operation for deep holes

For example, the multistep drilling operation as shown in is for deep holes with small diameters. The attributes and heritance in STEP-NC is shown in Figure 7.20. The programming format of multistep drilling cycle from Fanuc:

G83 X_Y_Z_R_Q_F_K

Where:

X_Y_: hole position

Z: Distance from point R to the bottom of the hole

R_:	Distance from initial level to point R
Q_:	depth of each cut
F_:	cutting feedrate
I_:	Forward and backward traveling speed
K_:	number of times the operation is repeated
P_:	dwel time at the bottom of the hole

The position of the hole is specified by X_Y_ parameter, which is used to create the attribute of *feature_placement* of the feature (hole). The parameter of Z_ is used to identify the depth of the hole. As mentioned earlier, the *round_hole* feature recognition is different. Here the feature recognition or identification of the feature geometry is realised here with, not before, the generation of its operation entity. The parameter R is equivalent to the *retract_plane*. F_ and I_ are used to specify the machining and retract feedrate, the corresponding entities of which in STEP-NC is *feedrate* (one attribute of *Technology*) and *feed_on_retract*. P_ is the duration of a dwell performed at the bottom of the hole, which can be accommodated in STEP-NC as *dwell_time_step*.

It is quite straightforward to translate these drilling cycle operations into the STEP-NC entity. Except the parameters addressed, the machine function and technology for the drilling type operation can be derived from miscellaneous commands of part programmes in a similar way for milling operations. The drilling cycle strategy is not that complex as the milling operation since the cutting strategy is already included in different types of drilling cycles. As shown in Figure 7.21, the multistep drilling strategy is used to drill a deep hole. The strategy definition of STEP-NC is:

```
ENTITY drilling_type_strategy;  
  reduced_cut_at_start:      OPTIONAL positive_ratio_measure;  
  reduced_feed_at_start:    OPTIONAL positive_ratio_measure;  
  depth_of_start:           OPTIONAL length_measure;  
  reduced_cut_at_end:       OPTIONAL positive_ratio_measure;  
  reduced_feed_at_end:      OPTIONAL positive_ratio_measure;  
  depth_of_end:             OPTIONAL length_measure;  
END_ENTITY;
```

As seen from the definition, the *drilling_type_strategy* is about the reduced cutting speed at the beginning and end of the operation as percentages of the programmed values. However, in NC part programmes there are no commands to address these parameters.

In practical machining, it is possible for the operator to control the cutting speed manually on the machine according to their experience. Since there are no records about this, it is difficult to capture this knowledge automatically. However it is possible to capture it through manually input by the shopfloor operators.

7.4. Standards compliant representation of resource independent process plan

In STEP-NC, there are mainly three types of information namely task description, features and workpiece geometry, operations (including cutting tools). To generate a STEP-NC process plan from part programmes, it is necessary to address how to produce these three types of information. The organisation of these three types of information in a STEP-NC file is shown in Figure 7.22. It starts with one top-level entity called project, which has the attributes to indicate the job related information: id, owner, release date and approval status etc. The main items describing the job are the workplan and the workpiece. The workplan contains information about the setup and the how to machine the part. The workpiece is the description of the raw material to start with, which is associated with workpiece_setup in the workplan. The majority of the workplan is its workingsteps. They are connected with the features and corresponding operations.

In Table 7.3, a comparison between STEP-NC and part programmes are presented in terms of information types and the method to close the information gaps are given. In CNC part programmes, there is no workplan, workpiece, setup and cutting tool information at all. There is information about features, operations and workingsteps, which can be recognised from part programmes. The workpiece including its setup information and cutting tool information are the inputs of process comprehension. The methods to recognise features and operations have been discussed in previous sections in this chapter. Then using the features and operations, the workingsteps can be generated. Based on all of above information, the project and workplan can be created although some information needs to be input by the users such as project description attributes.

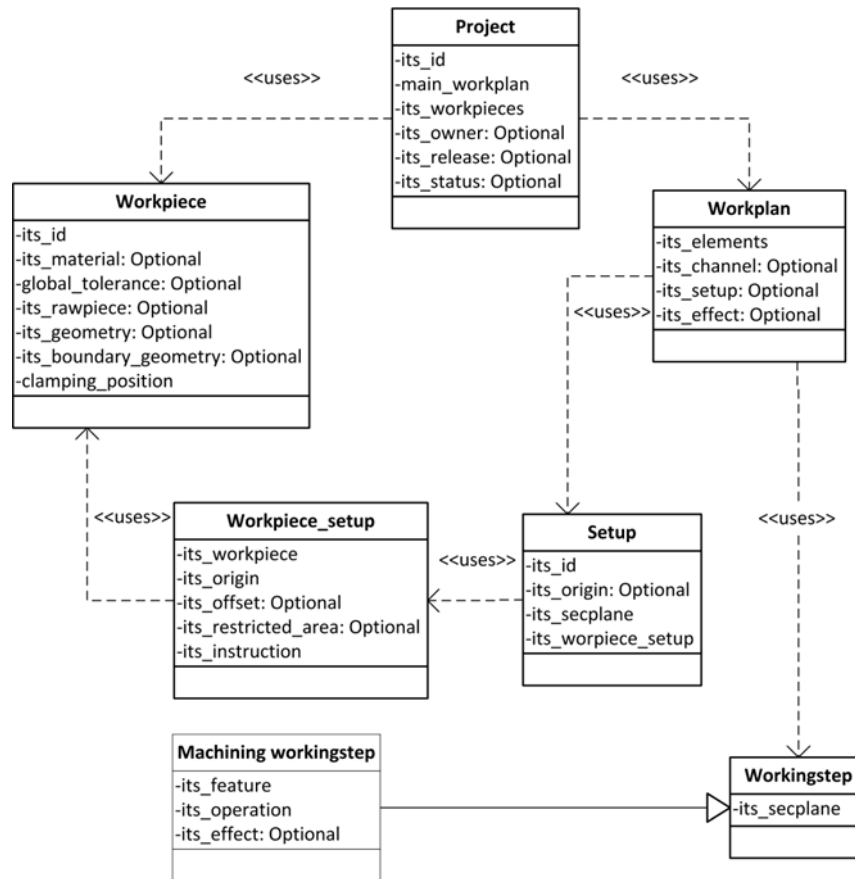


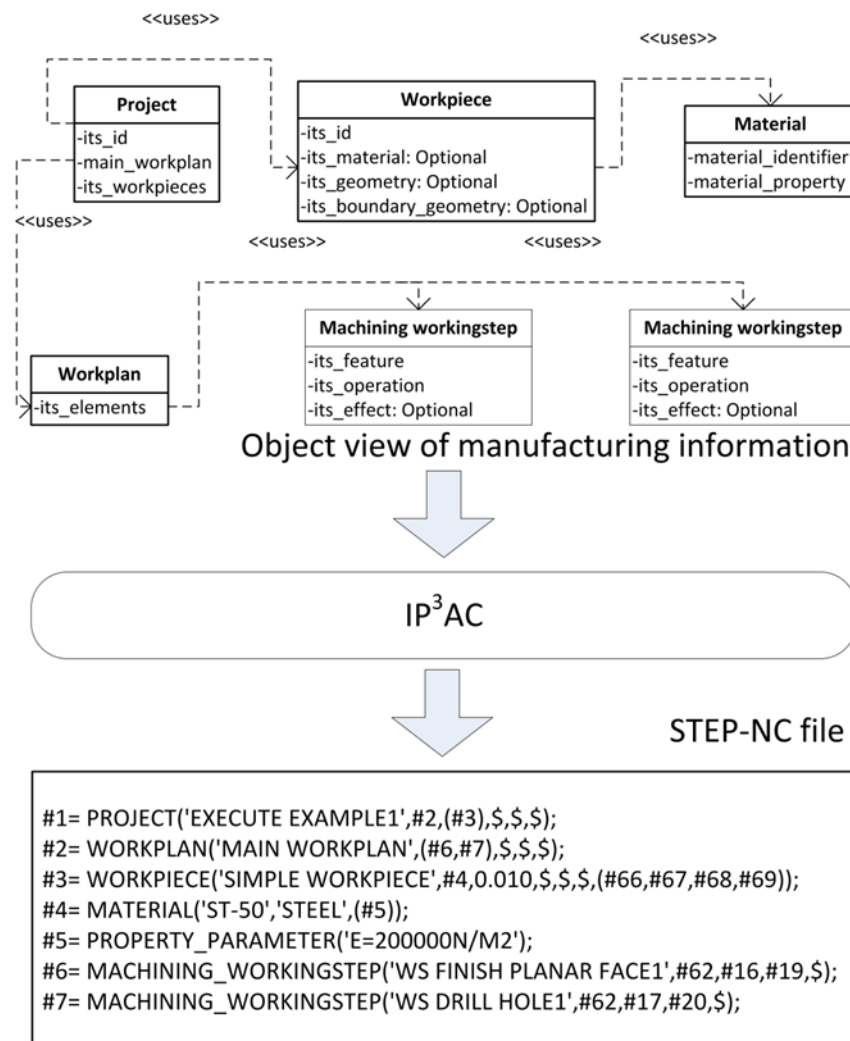
Figure 7.22 - The top level structure of STEP-NC data

Table 7.3 - STEP-NC versus part programme in terms of information types

STEP-NC entities	Part programmes	Generation method
Project	✗	Creation
Workplan	✗	Creation
Workpiece	✗	Input by user
Setup	✗	Input by user
Workingstep	✓	Recognition
Features	✓	Recognition
Operations	✓	Recognition
Cutting tool	✗	Input by user

Based on the entities recognised or created, a previously developed object-oriented library called IP³AC (Nassehi *et al.* 2007) is used to manipulate STEP-NC data and generate the STEP-NC representation, as shown in Figure 7.23. IP³AC is developed with Java and it “provides encapsulation of STEP-NC data in objects and by defining collections

of objects allows a JAVA program to manipulate manufacturing information in data structures that are native to the programming language. (Nassehi *et al.* 2007) Its Java-friendly characteristics and comprehensive access interfaces make it a perfect tool for this research to generate STEP-NC data. The relationship between IP³AC and UPCi is shown in Figure 7.24. UPCi reads in part programmes and recognises manufacturing knowledge. IP³AC organises manufacturing knowledge in STEP-NC format, stores it in Java objects and provides object-oriented access through interfaces. The output interface of IP³AC can generate STEP-NC text files. Both IP³AC and UPCi are running on the Java Virtual Machine (JVM), which provides an environment to execute Java programs. JVM is a piece of software which can be implemented on non-virtual hardware or standard operating systems (Lindholm and Yellin 1999). Hence, it enables a platform-independence scenario of “write once, run anywhere”.

Figure 7.23 - Using IP³AC to generate STEP-NC files

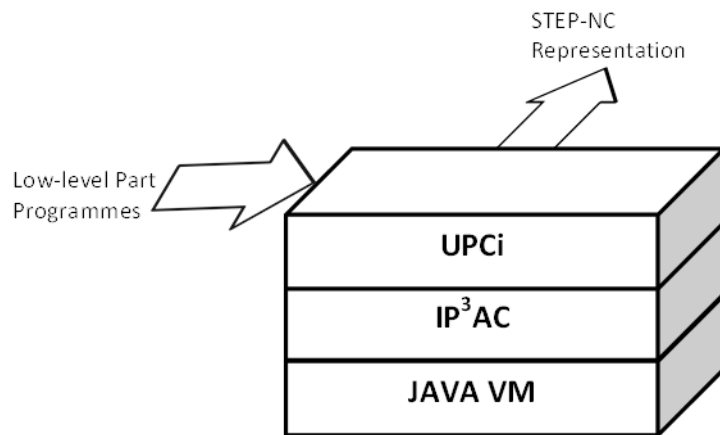


Figure 7.24 - UPi and IP³AC

7.5. Summary

In this chapter, the feature recognition, shopfloor knowledge capture and generation of STEP-NC representation of process plan are discussed. The logic of feature recognition from part programmes is presented. Several common features of prismatic parts are used as examples to validate the proposed recognition method. Two critical issues including interacting features recognition and feature identification are analysed and addressed. In the second part, the recognition of operations to capture the shopfloor knowledge is discussed in terms of drilling and milling types of operation. Finally, the STEP-NC generation method based on recognised features and operations is introduced.

8. Implementation of a software prototype for the UPCI

8.1. Introduction

A software prototype titled UPCI has been realised based on the framework outlined in Chapter 5. In this chapter, the development process is outlined using the key technologies introduced in Chapter 6 and Chapter 7.

8.2. An overview of the UPCI prototype system

The prototype of UPCI has the primary functions specified for the research framework in Chapter 5. This prototype converts part programmes into a standardised process plan in STEP-NC format, as shown in Figure 8.1. It realises an information flow from the shopfloor to the systems in the manufacturing network. In this prototype, two programming dialects used on two commercial controllers, a Fanuc controller and a Siemens controller, have been implemented as shown in Figure 8.2.

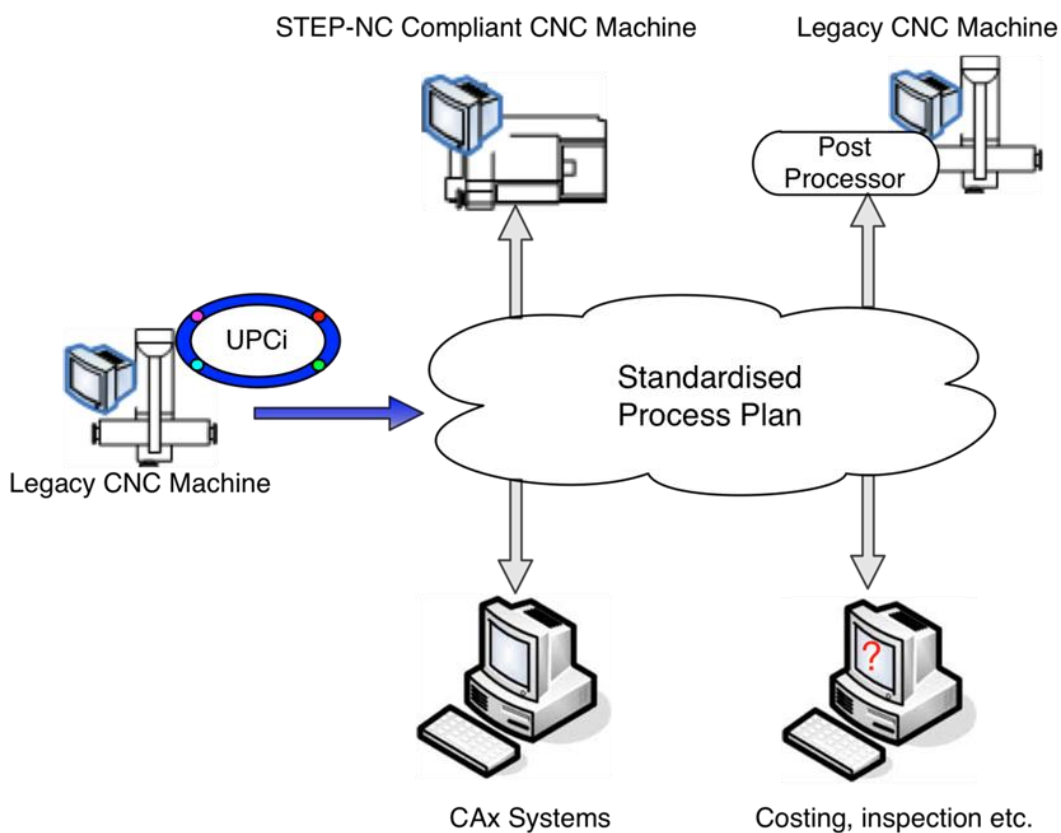


Figure 8.1 - UPCI prototype enables information flow from CNC machines to the manufacturing network

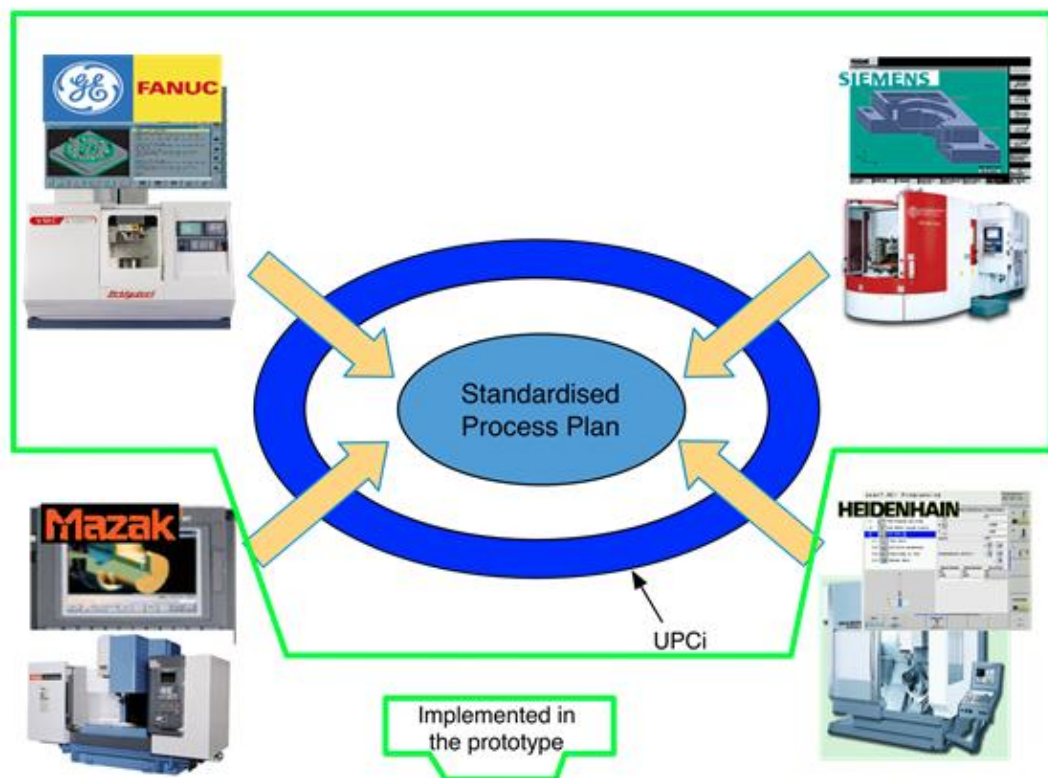


Figure 8.2 - Two programming dialects supported by the UPCI prototype

The UPCI prototype system has been developed with the object-oriented computer programming language Java, in an integrated development environment NetBeans (Version 6.8) (Oracle Corporation 2011). This is a user toolkit to develop desktop and internet based applications from the Oracle Corporation. An overview of the prototype system is shown in Figure 8.3.

In Figure 8.3, the menus in the red rectangle indicate the basic functions of the prototype system. The system accepts various G&M codes provided that the corresponding XML file of their programming specifications are loaded into the system. UPCI can check the part programmes against the XML specification, or vice versa. The '3D' menu is used to evoke the 3D viewer which can provide a 3D view of the toolpath of the part programme. It is helpful for the user to understand the part programme and check whether the part programme is translated correctly into the Meta-model. Since one of the inputs to UPCI is workpiece information, the 'workpiece' menu is used to call an input interface for the user to input workpiece geometry information and setup information. The next stage under the 'FR' menu is the feature recognition and shopfloor knowledge capture. They are implemented together since they are closely related. For machining strategy

recognition, a promoting window is used to ask the user to identify the machining strategy. At the last stage, IP³AC is used to generate the STEP-NC text file. In the main window of UPCI, the big text field on the right is showing the loaded text files and generated STEP-NC file. The 'Header' menu is provided for the user to tailor the header section of the STEP-NC file.

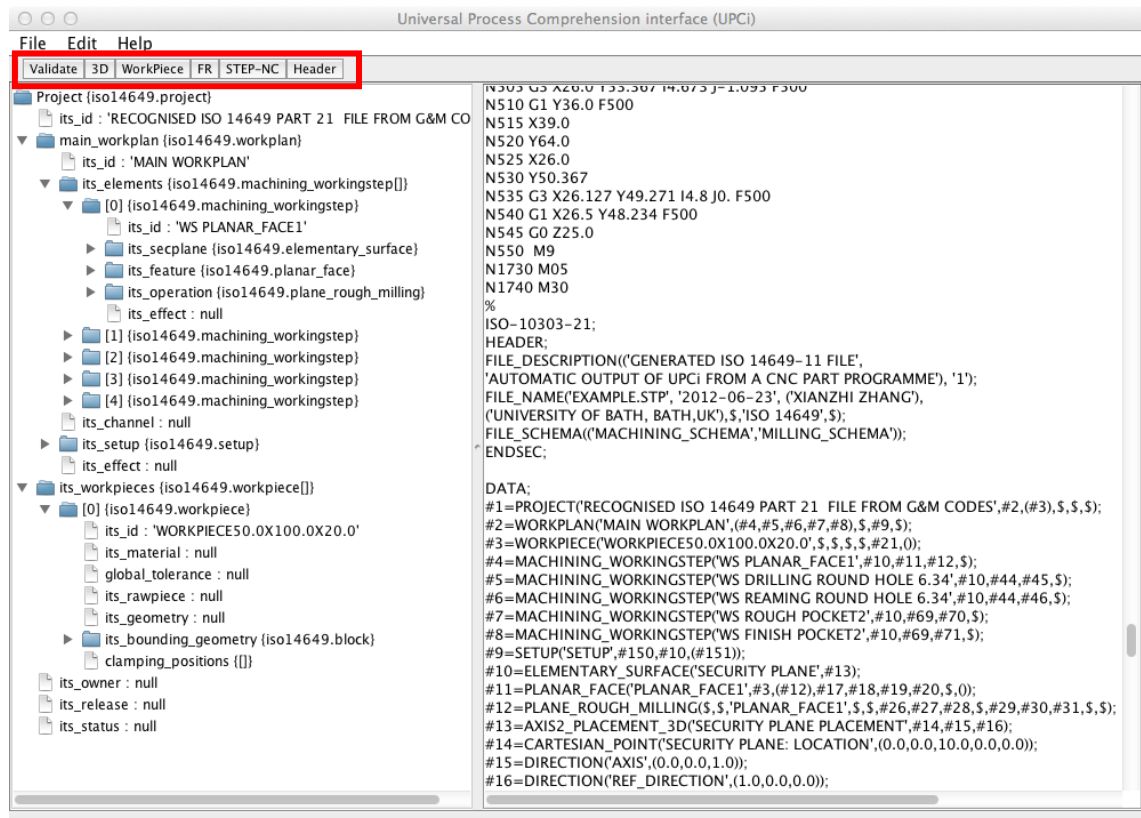


Figure 8.3 - Overview of UPCI software prototype

This implementation of UPCI fulfils the design considerations stated in Chapter 5. Implementation of UPCI does not need to make any changes to the current manufacturing resources. Secondly, it does support various dialects of programming languages. Its dictionary structure enables it to be expandable to support other programming dialects. Furthermore, UPCI has the ability to capture shopfloor knowledge and STEP-NC provides necessary accommodation mechanism for it. Since STEP-NC is a standardised representation of the process information, it is possible for other CAD/CAM or CNC machines to load and reuse it to reach an interoperable manufacturing scenario.

8.3. Development of the Meta-model of programming dialects

Based on the Meta-Model developed in Chapter 6, a UML model has been developed as shown in Figure 8.4. Each of the CNC commands is a sub-class of the root *GMIInstruction*, which defines common attributes and operation functions including commands line number, commands text, finding and setting commands' keywords etc. Sub-classes have their own specific attributes and functions to represent various CNC commands. This model doesn't cover all commands used on a CNC milling machines, for example the *DrillingCycle* in the diagram only has four types of drilling operation included. However, it is effortless to add more CNC commands into this model. For example, in implementation of the translation interface, it is easy to create a new class extended from the *DrillingCycle* to support, for instance, *Counterboring*, which is similar to a *boring* operation.

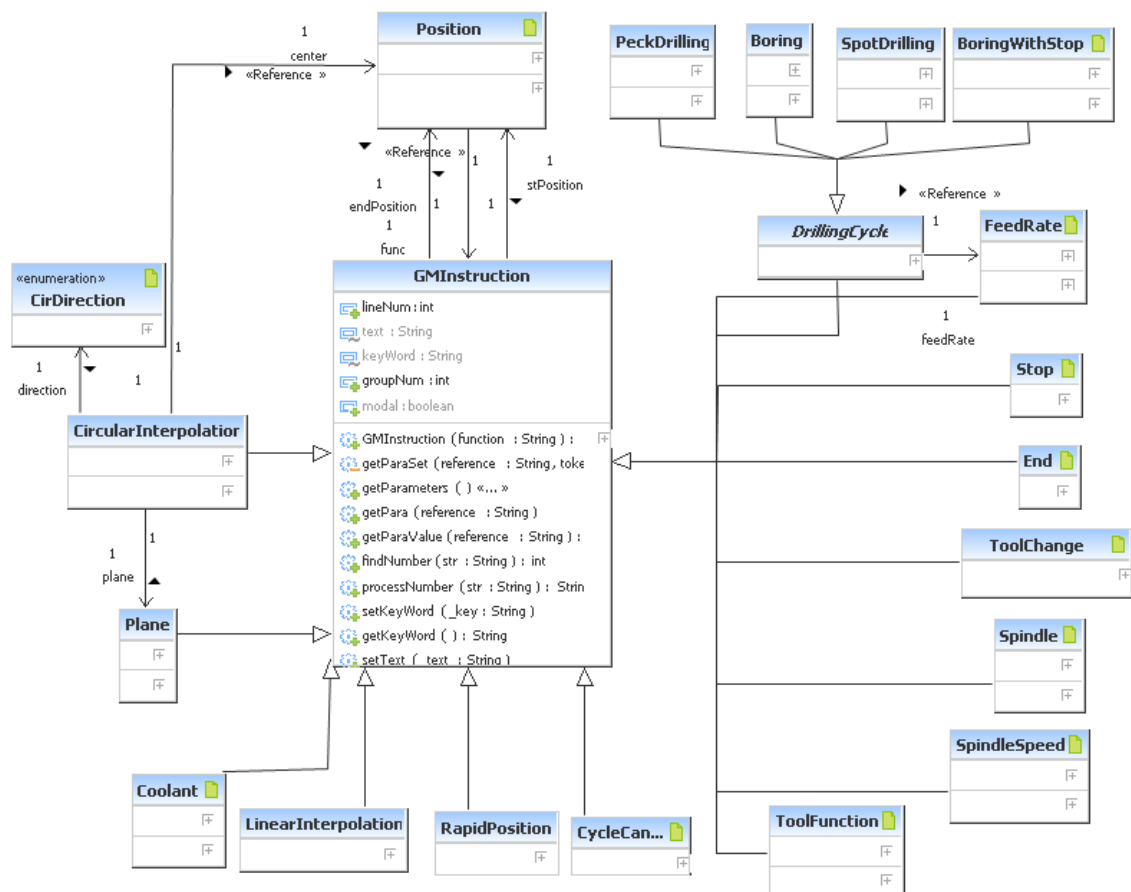


Figure 8.4 - Meta-Model of CNC programming languages

The UML model is then used to generate the corresponding Java class. These classes can then be utilised to instantiate various objects to represent the information carried by dialect-based part programmes. Hence, the next stage is to translate the part programmes into these objects (the Meta-model).

8.4. Translation from programming dialects into the Meta-model

As mentioned in Chapter 6, to translate programming dialects into the Meta-model, an XML description method has been used. It works as the translation reference from programming dialects into the Meta-model. Based on the XML schema, XML descriptions of programming dialects of G&M codes used on Fanuc 18i and Siemens 840D have been developed (see Appendix B). Using the XML descriptions, the part programmes can be translated into the Meta-model. As shown in Figure 8.5, a part adapted from ISO 14649 part 11 is designed in FeatureCAM, a commercial CAD/CAM system (Delcam 2012). To machine the part, there are four operations used to machine three features: a planar face, a rectangular pocket and a round hole. The detailed process plan is shown in Table 8.1. Under this process plan, G&M codes are post processed compatible with Fanuc controller (18i). The part programme is shown in Figure 8.6. In the following sections of this chapter, this sample part will be used to demonstrate the implementation of UPCi.

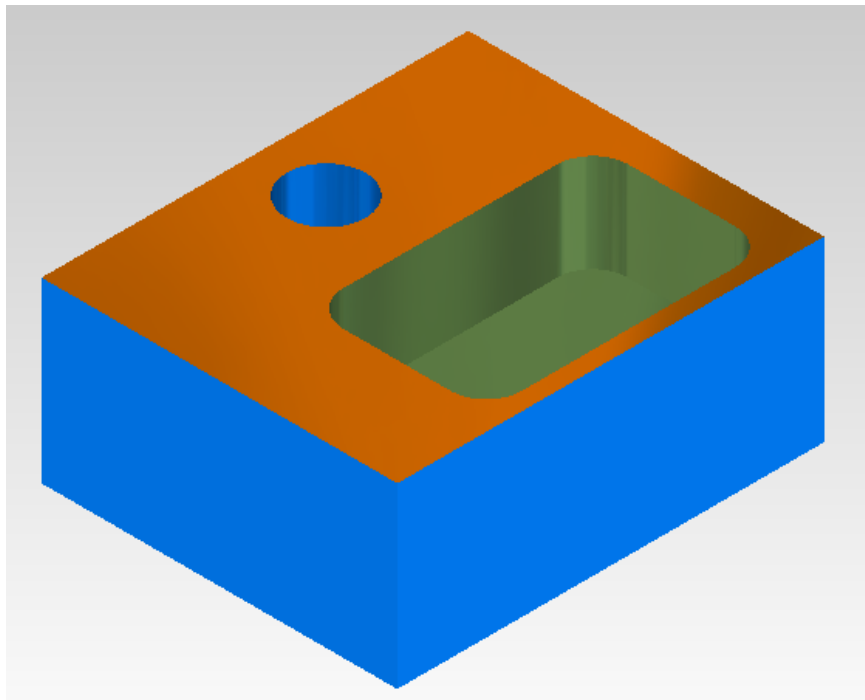


Figure 8.5 - Part designed in CAM system

Table 8.1 - Process plan sheet

Part	L 120.000 mm W 100.000 mm T 52.500 mm	ALUMINUM, 111.00 Brinell, 0.82 kN/mm ²		
OP1	face1 (face)	Tool: #1 (facemillM3200, 32.000 mm)	F/S: 10000 RPM, 5715 MPPM	Depth: 2.500 mm
OP2	rect_pock1 (rough1)	Tool: #2 (endmillM2000: 20.000 mm)	F/S: 3153 RPM, 2523 MPPM	Depth: 30.000 mm (in 2 steps, 15.000 mm each)
OP3	rect_pock1 (finish)	Tool: #2 (endmillM2000: 20.000 mm)	F/S: 4851 RPM, 2329 MPPM	Depth: 30.000 mm
OP4	hole1 (drill: Pecks: 2, Cycle: Deep Hole)	Tool: #3 (drill2000, 20.000 mm)	F/S: 1212 RPM, 0.300 MPPR	Centre: 60.000 mm 80.000 mm 0.000 mm Depth: 36.009 mm

%	N185 X74.57 Z-	19.98	
O0001(POCKET 11-4-	3.31	N335 X74.57 Z-	
2011)	N190 X54.57 Z-	21.65	N470 G3 X69.597
(Fadal Fanuc	4.98	N340 X54.57 Z-	Y15.0 I1.854
Operation: SETUP1)	N195 X74.57 Z-	23.32	J5.178 F1164.
(STOCK-DIMS)	6.65	N345 X74.57 Z-	N475 G1 X100.0
(TOOL-LIST)	N200 X54.57 Z-	24.99	F2328.
N40 G40 G80 G90	8.32	N350 X54.57 Z-	N480 Y45.0
G94	N205 X74.57 Z-	26.66	N485 X40.0
N45 M6T1	9.99	N355 X74.57 Z-	N490 Y15.0
N50 G0 G17 G54	N210 X54.57 Z-	28.33	N495 X72.597
X136.0 Y9.8 S10000	11.66	N360 X54.57 Z-	N500 G3 X74.451
M3	N215 X74.57 Z-	30.0	Y15.322 I0. J5.5
N55 G43 H1 Z25.0	13.33	N365 X85.43	F1164.
M8	N220 X54.57 Z-	F2522.	N505 G1 X76.088
N60 G8 P1	15.0	N370 Y30.43	Y16.25 F2328.
N65 Z3.0	N225 X85.43	N375 X54.57	N510 G0 Z25.0
N70 G1 Z-2.5 F5000.	F2522.	N380 Y29.57	N515 M9
N75 X-16.0	N230 Y30.43	N385 G3 X58.958	N520 G53 Z0.
N80 Y35.6	N235 X54.57	Y23.851 I5.921 J0.	N525 M5
N85 X136.0	N240 Y29.57	N390 X66.105	N530 (DRILL
N90 Y61.4	N245 G3 X58.958	Y22.91 I7.147	HOLE1)
N95 X-16.0	Y23.851 I5.921 J0.	J26.673	N535 G40 G80
N100 Y87.2	N250 X66.105	N395 G1 X92.09	G90 G94
N105 X136.0	Y22.91 I7.147	N400 Y37.09	N540 M6T3
N110 G0 Z25.0	J26.673	N405 X47.91	N545 M1
N115 G8 P0	N255 G1 X92.09	N410 Y22.91	N550 G0 G17 G54
N120 M9	N260 Y37.09	N415 X54.57	X60.0 Y80.0 S1212
N125 G53 Z0.	N265 X47.91	N420 G2 X60.338	M3
N130 M5	N270 Y22.91	Y19.58 I0. J-6.66	N555 G43 H3
N135	N275 X54.57	N425 G3 X66.105	Z25.0 M8
(RECTANGULAR	N280 G2 X60.338	Y16.25 I5.768 J3.33	N560 G83 G98
POCKET	Y19.58 I0. J-6.66	N430 G1 X98.75	R3.0 Z-36.009
RECT_POCK1)	N285 G3 X66.105	N435 Y43.75	Q20.0 F364.
N140 G40 G80 G90	Y16.25 I5.768	N440 X41.25	N565 G0 G80
G94	J3.33	N445 Y16.25	Z25.0
N145 M6T2	N290 G1 X98.75	N450 X66.105	N570 M9
N150 M1	N295 Y43.75	N455	N575 G53 Z0.
N155 G0 G17 G54	N300 X41.25	(RECTANGULAR	N580 M5
X74.57 Y29.57 S3153	N305 Y16.25	POCKET FINISH	N585 G0G90G53
M3	N310 X66.105	RECT_POCK1)	X0. Y10.
N160 G43 H2 Z25.0	N315 X74.57	N460 G1 G17 G94	N590 M6T1
M8	Y29.57	G54 X66.105	N595 M30
N165 G8 P1	N320 X54.57 Z-	Y16.25 Z-30.0	%
N170 Z3.0	16.64 F1261.	S4851	
N175 G1 Z0.03	N325 X74.57 Z-	N465 X67.742	
F1261.	18.31	Y15.322 F2328.	
N180 X54.57 Z-1.64	N330 X54.57 Z-		

Figure 8.6 - G&M codes for the sample part

Using the Fanuc programming syntax description file, the part programmes can be translated in the Meta-model. After the translation, all the information is stored in an inside array of UPCi. Based on that, it is possible to visualise them to validate the translation. As shown in Figure 8.7, the toolpath of the part programme input into the system can be viewed. It provides a 3D multi-angle visualisation. It is helpful to identify the tools, operation and strategies in the following process comprehension activities.

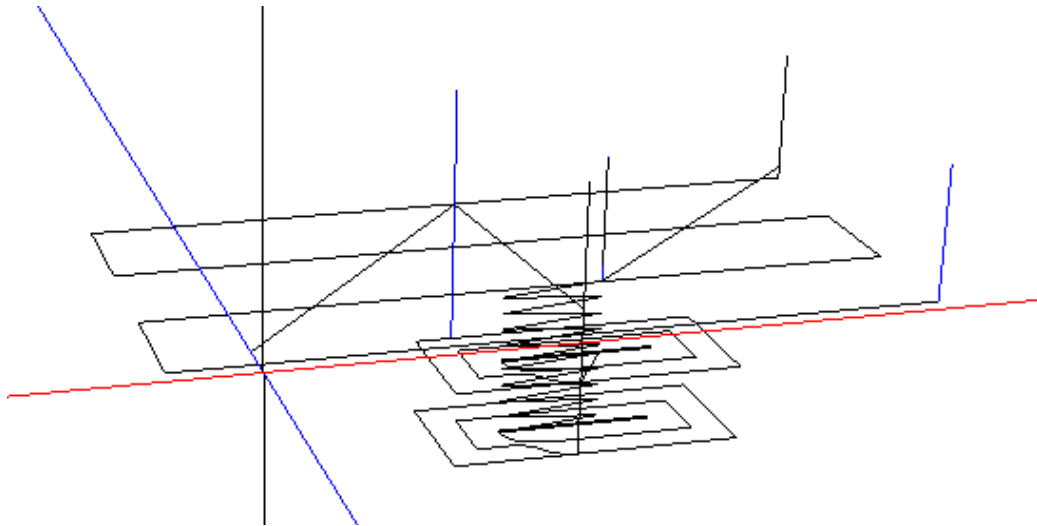


Figure 8.7 - Toolpath of the part programme

8.5. Feature recognition and knowledge capture

After the part programmes have been read into the system, the next step is using the information in a Meta format to recognise the features and capture the manufacturing knowledge inside the part programmes. To input the cutting tool information and workpiece data including its setup information, UPCi provides user interfaces to enter those sets of information. When the feature recognition function is activated, the promoting windows are presenting to receive necessary information from the user. In Figure 8.8, the window is receiving workpiece definition and its setup information. In Figure 8.9, the user is required to enter the cutting tool information for that part programme section.

In this prototype system of UPCi, the workpiece definition window provides a method to define a piece of box or block shaped raw material. In STEP-NC, there are three entities to define a workpiece: block, cylinder or define its geometry. Block and cylinder is based on the geometric items of ISO 10303-42, which specifies the resource constructs for the explicit geometric and topological representation of the shape of a product (ISO 10303-

42 2003). When a complex shaped workpiece is needed, an entity from ISO 10303-514 (ISO 10303-514 1994) is used, named `advanced_brep_shape_representation`. Since in manufacturing of a prismatic part with milling technology a block is most frequently used, a block definition is provided in the development of this prototype system. Another issue about the workpiece is its setup. Since technically the raw workpiece can be anywhere inside of the machining table as long as it can be reached by the cutting tool, the programming is executed by the machine after the operator defines a setup coordinate system. All of the coordinates in a piece of part programme are based on this setup coordinate system. Hence, in the workpiece definition window, a 3D view of the toolpath is shown in the setup coordinate system as shown in Figure 8.8. When a workpiece is defined, the position of the workpiece can be observed relative to the toolpath to assist the user and prevent potential mistakes to position the raw block.

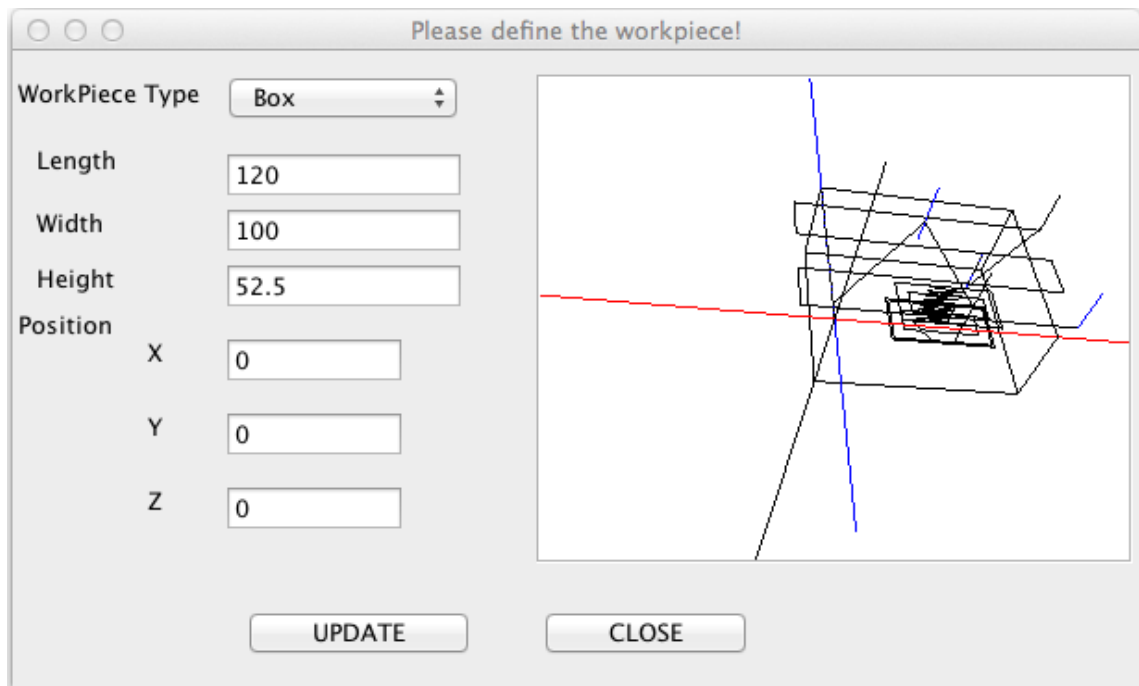


Figure 8.8 - Workpiece definition

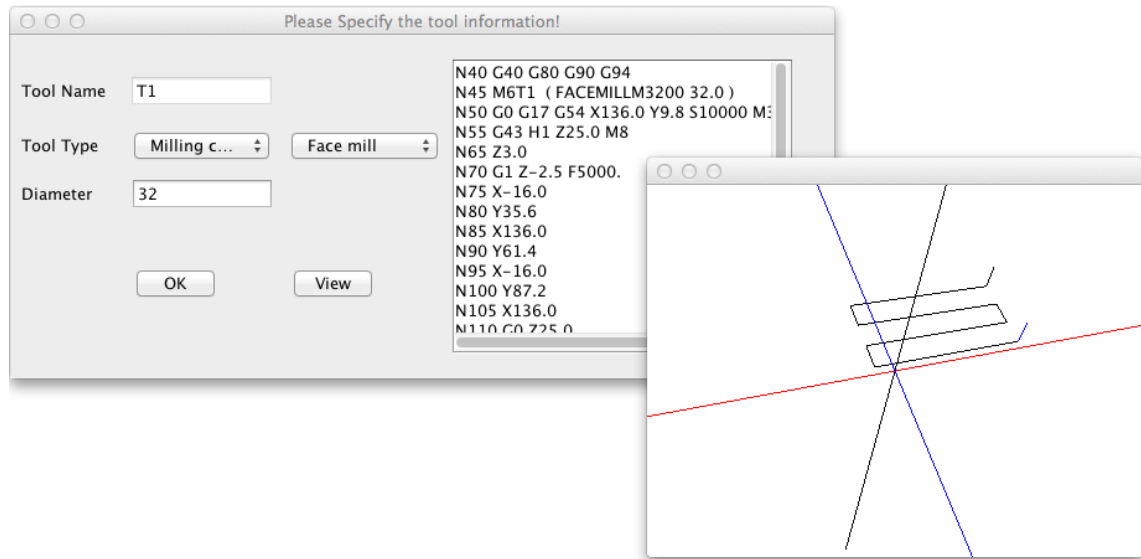





Figure 8.9 - Cutting tool definition

In the cutting tool definition window in Figure 8.9, to help the user acknowledge the current operation the tool is applied with, the section of the part programme is shown on the right. The comments in the part programme are helpful to the user to decide the tool and its diameter. In this figure, the comments “FACEMILLM 3200 32.0” remind the user the tool is facemill with the diameter of 32mm. If needed, the access to a 3D view of the toolpath for the section of part programme is provided. In the window, the tool name is from the part programme. In this window, the user only needs to specify the tool type and its diameter. Other attributes of the cutting tool is not so relevant to this research such as the cutting length of the tool or the number of the tool tips. Using the workpiece and cutting tool information, UPCi recognises features and decides on the operations using the method specified in Chapter 7. The recognised operations and features are listed in Table 8.2. There are four operations and corresponding to four features recognised. In the last column of Table 8.2, it is the covering area of the toolpath for each operation. Since there is no toolpath defined in the drilling canned cycle, the cutting area of drilling operation is not available in this case. It is worthy to note that this is not the final result of the feature recognition as discussed in Chapter 7. There is a further requirement to identify the redundant features since they do not exist in the original process plan.

Table 8.2 - Recognised operations and features

Operation	Operation type	Feature	Covering area of toolpath
OP1	Plane_milling	Planar face	
OP2	Bottom_and_side_milling	Pocket	
OP3	Bottom_and_side_milling	Pocket with a boss	
OP4	Drilling	Round hole	N/A

In Figure 8.10, the user is asked to tell the system what kind of machining strategy is used in an operation. This happens after the feature and operation types have been recognised. The user is provided with the feature, operation and tool information. Based on the toolpath view of the operation, the user can judge and choose the appropriate machining strategy and approach/retract strategies. Since the approach strategy and retract strategy are optional in STEP-NC, if the user is not sure or no appropriate options, the approach and retract strategies can be left empty. It is worthy to note that, if necessary for some complex operations and none of the strategies provided can describe the machining strategy, the `explicit_strategy` can be chosen to keep the toolpath and record it in STEP-NC files. It is helpful to keep the original process knowledge, but leaves less flexibility for the process plan in STEP-NC files by copying the original toolpath.

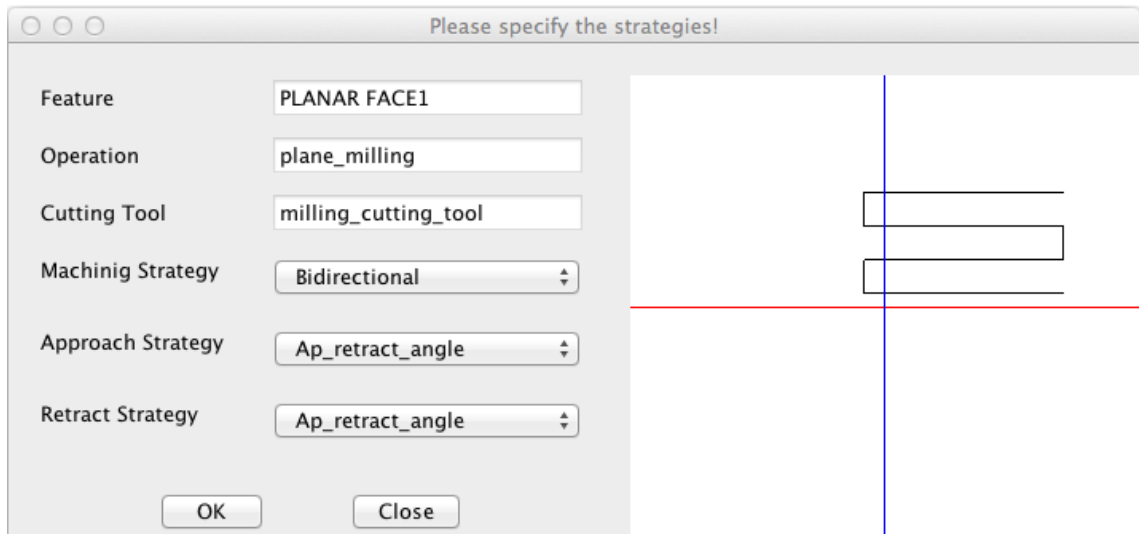


Figure 8.10 - Machining strategies

All of the manufacturing information extracted from part programme and created based on the user input is stored in a linear array in the system as objects of Java classes. These Java classes are generated based on the STEP-NC entities. This research is using a library of these Java classes developed by Nassehi et al.(2007). Together with the Java class library, they also developed interfaces to provided access and manipulating methods to STEP-NC data. The whole package including the Java library and access methods is called IP³AC. In Section 8.7, IP³AC is used to generate the STEP-NC presentation of process plan extracted from the part programmes.

8.6. Coordinate system update of STEP-NC data

Before the generation of final STEP-NC representation, there is a need to update the coordinate system used by recognised features to make sure the recognised features are at the right location to form the component correctly. The coordinate system used in STEP-NC is a Cartesian coordinate system as shown in Figure 8.11. There are several levels of coordinate system. The advantage of that is it is helpful to understand the part geometry more easily and alter the part geometry at a specific element. For example, if you want to change the location of a pocket in a part the only element that needs to be changed is the feature placement based on the workpiece setup coordinate system. There is no need to figure out the pocket geometry data. All the geometrical elements inside the pocket such as feature boundary, location of its boss(es) are in the feature coordinate system based on the feature placement in the workpiece coordinate system. It is the

same when the location of the whole workpiece is to be changed inside the setup coordinate system.

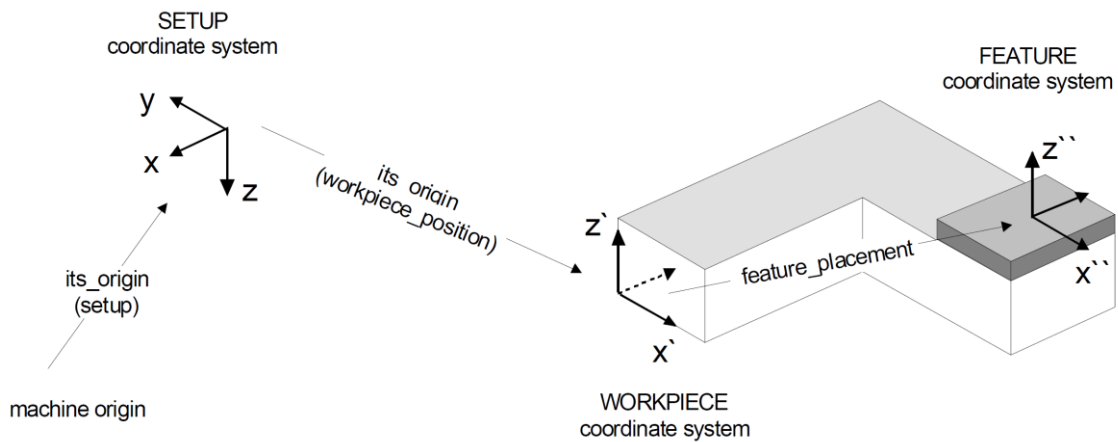


Figure 8.11 - STEP-NC coordinate system

In NC part programmes, all the coordinates (absolute or relative) are based on one single coordinate system used in the CAD/CAM system. In execution the operator needs to recreate the coordinate system on the machine based on the location of the workpiece fixed on the machine table. Now in this research, the coordinate system in the NC part programme is needed to be copied and translated into the STEP-NC coordinate systems. Taking the workpiece as the starting point, it is logical to take the coordinate system in NC part programmes as the setup coordinate system in STEP-NC. In CAD/CAM system, there is only one coordinate system. The workpiece and cutting toolpath are all based on it; while in process comprehension, the workpiece is defined based on the coordinate system of part programmes with the reference of cutting tool path, as shown in Figure 8.8. Hence, the coordinate system used in CAD/CAM can be taken as equivalent as the setup coordinate system in STEP-NC.

Under the setup coordinate system, there are usually two sub-coordinate systems: workpiece coordinate system and feature coordinate system. The workpiece coordinate system is based on the workpiece placement in the setup coordinate system and the feature coordinate system is based on the feature placement in the workpiece coordinate system. Consequently the geometry elements related with the workpiece needs one transformation into its own coordinate system from the CAD/CAM coordinates while the feature geometry data needs two transformations. For example, the feature depth is based on the feature coordinate system and it needs to be transformed from its original coordinates into the workpiece coordinate system and then into the feature coordinate

system. Technically, if there are further sub-coordinate systems under feature coordinate system, more transformations are needed. It happens when a feature has a sub feature (a boss) or a geometrical item (`general_closed_profile`) to identify the feature boundary, which has its own placement and coordinate system. However, to be straightforward, features have been identified as the prime unit of product geometry and all feature related geometrical elements, including sub-features, are all based on the feature placement.

In STEP-NC, the placement of the feature or a geometry item is a translation and/or a rotation which transforms the origin of the containing coordinate system origin into the origin of the feature's or the geometry item's local coordinate system. For a feature, the placement is usually (except bosses) relative to the workpiece coordinate system. It defines a new feature-local coordinate system by defining the new origin and axis directions. To define a new coordinate system in STEP-NC, an entity called `axis2_placement_3d` is used. The definition of it is as follows:

```
ENTITY placement
location : cartesian_point;
END_ENTITY;

ENTITY axis2_placement_3d
SUBTYPE OF (placement);
axis : OPTIONAL direction;
ref_direction : OPTIONAL direction;
END_ENTITY;
```

Location is a point with coordinates to identify the origin for the new coordinate system. Axis is used to define the new Z axis direction relative to the containing coordinate system. Ref_direction is the new X axis direction. The new Y axis can be identified using right-handed orientation rule. As shown in Figure 8.12, the origin and axis direction of the new coordinate system have been specified. The axis direction is defined by an array of three ratios such as (1, 0, 0), which means the direction of the X axis in the containing coordinate system.

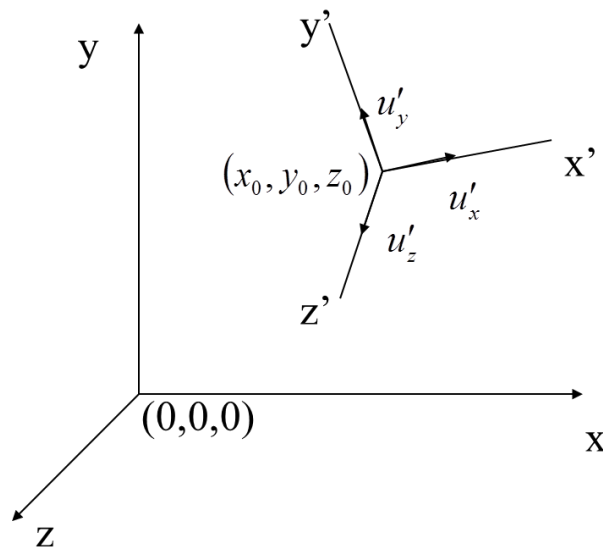


Figure 8.12 - Coordinates transformation

In the containing coordinate system, a point with the coordinates (x,y,z) needs to be transformed into a new coordinate system, whose origin is (x_0,y_0,z_0) and orientations of three axis x' , y' and z' are $(\mu'_{x1}, \mu'_{x2}, \mu'_{x3})$, $(\mu'_{y1}, \mu'_{y2}, \mu'_{y3})$ and $(\mu'_{z1}, \mu'_{z2}, \mu'_{z3})$. The coordinates of the point in the new coordinate system can be calculated through the transformation formula as follows:

$$(x', y', z', 1) = (x, y, z, 1) \cdot T \cdot R$$

Where T is the translation matrix and R is the rotation matrix:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x_0 & -y_0 & -z_0 & 1 \end{bmatrix}$$

$$R = \begin{bmatrix} u'_{x1} & u'_{y1} & u'_{z1} & 0 \\ u'_{x2} & u'_{y2} & u'_{z2} & 0 \\ u'_{x3} & u'_{y3} & u'_{z3} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

8.7. Generation of STEP-NC representation of process plan

The final stage of process comprehension is the generation of STEP-NC presentation of the process plan. The format of a STEP-NC file can be text based formalised as ISO 10303-21 (ISO 10303-21 2002) or XML based formalised as ISO 10303-28 (ISO 10303-28 2007). The text format, usually referred as a part 21 file or STEP physical file is more popular (Xu *et al.* 2005), since it is easy to read. In this research, a part 21 file has been chosen to be the final presentation of the process plan.

At this stage, the majority of the information required to produce a STEP-NC process plan has been recognised or acquired from the user. As mentioned in Chapter 7, to produce a complete STEP-NC file, there are two more entities needed namely project and workplan. “The entity workplan allows to combine several workingsteps and NC functions in a linear order”(ISO 14649-10 2002). It serves as an attribute of the top level entity project. It defines a linear sequence of operations to finish a part. Actually, there are several other structures available in STEP-NC to organise operations: parallel, non_sequential, selective, if_statement, while_statement, assignment. Since most of the part programmes used today are linear sequenced, a workplan is used to accommodate workingsteps, which are generated based on operations in the same sequence. The operations are the base of workingsteps and each operation needs to be associated with a workingstep. One or more operations are applied with a feature. As shown in Table 8.2, there is a feature for each operation. As mentioned in Chapter 7, sometime it is not correct since it is possible for a feature to have more than one operation. In UPCI, the user is offered to judge whether there should be a new feature or not for an operation in the cases of complex situations. If not, the operation should be associated with another recognised feature. For example, in Table 8.3 the OP2 and OP3 are the rough and finish operation for the same feature, a rectangular pocket. There is no such feature: a pocket with a boss as shown in Table 8.2. After removing the redundant features, workingsteps can be generated for each operation. As shown in Table 8.3, there are four workingsteps in correspondence with four operations.

Table 8.3 - STEP-NC process plan

Workingstep	Operation	Operation type	Feature
WS1	OP1	Plane_milling	Planar face1
WS2	OP2	Bottom_and_side_milling (Rough)	Pocket1
WS3	OP3	Bottom_and_side_milling (Finish)	Pocket1
WS4	OP4	Drilling	Round hole1

Project is the top level entity in a STEP-NC file, and usually a STEP-NC file starts with it. As shown in Figure 7.23, project contains the workpiece, workplan, which have been created and other description attributes: id, owner, release date and approval status. After project and workplan have been created, all manufacturing information are encapsulated in various objects and stored in an array in the system with linking references between them. Then IP³AC is used to convert the object view of manufacturing knowledge into part 21 text file. However, in a part 21 STEP-NC file, there are two sections: header and data. The entities discussed so far are all in the data section. IP³AC only produces the data section of the STEP-NC file since the header of the file is only a description of the file and depending on the situation. Hence, in UPCi, a template of the header is provided and the user can edit it according to their needs. A complete STEP-NC part 21 file generated for the part programme is shown in Figure 8.14.

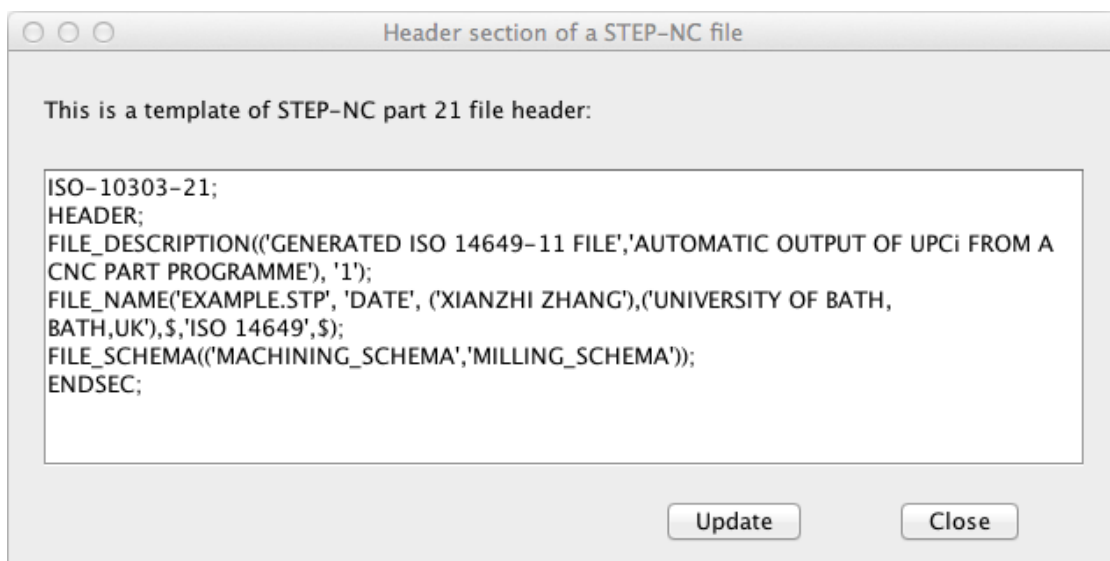


Figure 8.13 - Header template for STEP-NC part 21 files

```

ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('GENERATED FOR ISO 14649-11 EXAMPLE 1','SIMPLE PRORGRAM WITH A
PLANAR_FACE, A POCKET AND A ROUND_HOLE'),'1');
FILE_NAME('EXAMPLE2.STP','2011-11-03','(XIANZHI ZHANG)', ('UNIVERSITY OF BATH,
BATH','UK'),$, 'ISO 14649',$);
FILE_SCHEMA(('MACHINING_SCHEMA','MILLING_SCHEMA'));
ENDSEC;
DATA;
#1=PROJECT('RECOGNISED FROM G&M CODES',#2,(#3),$, $, $);
#2=WORKPLAN('MAIN WORKPLAN',(#4,#5,#6),$, #7,$);
#3=WORKPIECE('WORKPIECE120.0X100.0X50.0',$, $, $, $, #20,());
#4=MACHINING_WORKINGSTEP('WS PLANAR FACE1',#8,#9,#10,$);
#5=MACHINING_WORKINGSTEP('WS POCKET 77.5X47.5',#8,#45,#46,$);
#6=MACHINING_WORKINGSTEP('WS DRILLING ROUND HOLE 20.0',#8,#70,#71,$);
#7=SETUP('SETUP',#88,#8,(#89));
#8=ELEMENTARY_SURFACE('SECURITY PLANE',#11);
#9=PLANAR_FACE('PLANAR FACE1',#3,(#10),#15,#16,#17,#18,#19,());
#10=PLANE_MILLING($,$,'PLANAR FACE1',$, $, #25,#26,#27,$, $, $, $, $);
#11=AXIS2_PLACEMENT_3D('SECURITY PLANE PLACEMENT',#12,#13,#14);
#12=CARTESIAN_POINT('SECURITY PLANE: LOCATION',(0.0,0.0,10.0,0.0,0.0));
#13=DIRECTION('AXIS',(0.0,0.0,1.0));
#14=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#15=AXIS2_PLACEMENT_3D('PLANAR FACE1 PLACEMENT',#30,#31,#32);
#16=ELEMENTARY_SURFACE('PLANAR FACE1 DEPTH PLANE',#33);
#17=LINEAR_PATH($,#37,#38);
#18=LINEAR_PROFILE($,#39);
#19=RECTANGULAR_CLOSED_PROFILE(#40,#41,#37);
#20=BLOCK('WORKPIECE BLOCK',#21,120.0,100.0,50.0);
#21=AXIS2_PLACEMENT_3D('WORKPIECE BLOCK PLACEMENT',#22,#23,#24);
#22=CARTESIAN_POINT('WORKPIECE BLOCK: LOCATION',(0.0,0.0,0.0));
#23=DIRECTION('AXIS',(0.0,0.0,1.0));
#24=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#25=MILLING_CUTTING_TOOL('T1',#28,(),$, $, $);
#26=MILLING_TECHNOLOGY(5000.0,TCP.,$, 10000.0,$, $, $, $);
#27=MILLING_MACHINE_FUNCTIONS(.F.,$, $, $, $, $, $, $, $);
#28=FACEMILL(#29,$, $, $, $);
#29=MILLING_TOOL_DIMENSION(32.0,$, $, $, $, $);
#30=CARTESIAN_POINT('PLANAR FACE1',(60.0,48.5,-2.5));
#31=DIRECTION('AXIS',(0.0,0.0,1.0));
#32=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#33=AXIS2_PLACEMENT_3D('PLANAR FACE1 DEPTH',#34,#35,#36);
#34=CARTESIAN_POINT('PLANAR FACE1 DEPTH',(76.0,-38.7,0.0));
#35=DIRECTION('AXIS',(0.0,0.0,1.0));
#36=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#37=TOLERANCED_LENGTH_MEASURE(184.0,$);
#38=DIRECTION('PLANAR FACE DIRECTION',(0.0,1.0,0.0));
#39=NUMERIC_PARAMETER('null',109.4,'mm');
#40=AXIS2_PLACEMENT_3D('PLANAR FACE1 PROFILE PLACEMENT',#42,#43,#44);
#41=TOLERANCED_LENGTH_MEASURE(109.4,$);
#42=CARTESIAN_POINT('PLANAR FACE1 PROFILE LOACTION',(60.0,48.5,-2.5));
#43=DIRECTION('AXIS',(0.0,0.0,1.0));
#44=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#45=CLOSED_POCKET('POCKET 77.5X47.5',#3,(#46),#47,#48,(),$, #49,$, #50,#51);
#46=BOTTOM_AND_SIDE_MILLING($,$,'POCKET 77.5X47.5',$, $, #52,#53,#54,$, $, $, $, $, $);

```

```

#47=AXIS2_PLACEMENT_3D('POCKET 77.5X47.5 PLACEMENT',#57,#58,#59);
#48=ELEMENTARY_SURFACE('POCKET 77.5X47.5 DEPTH PLANE',#60);
#49=PLANAR_POCKET_BOTTOM_CONDITION();
#50=TOLERANCED_LENGTH_MEASURE(10.0,$);
#51=RECTANGULAR_CLOSED_PROFILE(#64,#65,#66);
#52=MILLING_CUTTING_TOOL('T2',#55,(),$,$,$);
#53=MILLING_TECHNOLOGY(2522.0,.TCP,,$,3153.0,$,$,$,$);
#54=MILLING_MACHINE_FUNCTIONS($,$,$,$,$,(),$,$,$,());
#55=ENDMILL(#56,$,$,$,$);
#56=MILLING_TOOL_DIMENSION(20.0,$,$,$,$,$);
#57=CARTESIAN_POINT('POCKET 77.5X47.5 LOCATION',(70.0,30.0,-15.0));
#58=DIRECTION('AXIS',(0.0,0.0,1.0));
#59=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#60=AXIS2_PLACEMENT_3D('POCKET 77.5X47.5 DEPTH',#61,#62,#63);
#61=CARTESIAN_POINT('POCKET 77.5X47.5 DEPTH',(-15.43,-0.4299999999999997,-15.0));
#62=DIRECTION('AXIS',(0.0,0.0,1.0));
#63=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#64=AXIS2_PLACEMENT_3D('POCKET PROFILE PLACEMENT',#67,#68,#69);
#65=TOLERANCED_LENGTH_MEASURE(47.5,$);
#66=TOLERANCED_LENGTH_MEASURE(77.5,$);
#67=CARTESIAN_POINT('POCKET PROFILE LOCATION',(70.0,30.0,-15.0));
#68=DIRECTION('AXIS',(0.0,0.0,1.0));
#69=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#70=ROUND_HOLE('ROUND HOLE 20.0',#3,(#71),#72,#73,#74,$,$);
#71=MULTISTEP_DRILLING($,$,'DRILLING ROUND HOLE
20.0',$,$,#75,#76,#77,$,$,$,$,#78,0.0,20.0,20.0,$);
#72=AXIS2_PLACEMENT_3D('ROUND HOLE 20.0 PLACEMENT',#81,#82,#83);
#73=ELEMENTARY_SURFACE('ROUND HOLE 20.0 DEPTH PLANE',#84);
#74=TOLERANCED_LENGTH_MEASURE(20.0,$);
#75=MILLING_CUTTING_TOOL('T3',#79,(),$,$,$);
#76=MILLING_TECHNOLOGY(364.0,.TCP,,$,1212.0,$,$,$,$);
#77=MILLING_MACHINE_FUNCTIONS(.F,,$,$,$,$,(),$,$,$,());
#78=DRILLING_TYPE_STRATEGY($,$,$,$,$);
#79=TWIST_DRILL(#80,$,$,$,$);
#80=MILLING_TOOL_DIMENSION(20.0,$,$,$,$,$);
#81=CARTESIAN_POINT('ROUND HOLE 20.0 LOCATION',(60.0,80.0,0.0));
#82=DIRECTION('AXIS',(0.0,0.0,1.0));
#83=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#84=AXIS2_PLACEMENT_3D('ROUND HOLE 20.0 DEPTH',#85,#86,#87);
#85=CARTESIAN_POINT('ROUND HOLE 20.0 DEPTH',(60.0,80.0,-14.009));
#86=DIRECTION('AXIS',(0.0,0.0,1.0));
#87=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#88=AXIS2_PLACEMENT_3D('SETUP ORIGIN',#90,#91,#92);
#89=WORKPIECE_SETUP(#3,#93,$,$,());
#90=CARTESIAN_POINT('SETUP LOCATION',(0.0,0.0,0.0));
#91=DIRECTION('AXIS',(0.0,0.0,1.0));
#92=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#93=AXIS2_PLACEMENT_3D('WORKPIECE120.0X100.0X50.0 SETUP',#94,#95,#96);
#94=CARTESIAN_POINT('WORKPIECE120.0X100.0X50.0SETUP LOCATION',(0.0,0.0,0.0));
#95=DIRECTION('AXIS',(0.0,0.0,1.0));
#96=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
ENDSEC;

```

Figure 8.14 - STEP-NC representation of the process plan

8.8. Prototype integration

Based on the functions in previous sections of this chapter, a prototype integration of UPCI has been implemented to function as a single system. Figure 8.15 shows a class diagram for the major components of UPCI.

There are mainly three packages in the system: *GeneralNC*, *ProcessComprehension* and *ISO14649*. *GeneralNC* is the class library of the Meta-model of CNC programming languages. *ISO14649* is the class library from IP³AC. *processComprehension* is the core of UPCI, which takes the Meta-model data of part programmes to generate STEP-NC data of the process plan. Only some of the key classes are listed in the diagram. The *ProcessComprehensionFrm* is the main user interface that hosts other viewers or dialog windows. It creates objects from other classes which realise the process comprehension activities. With the help of the user, this prototype of UPCI can realise the process comprehension to convert the part programmes into STEP-NC representations.

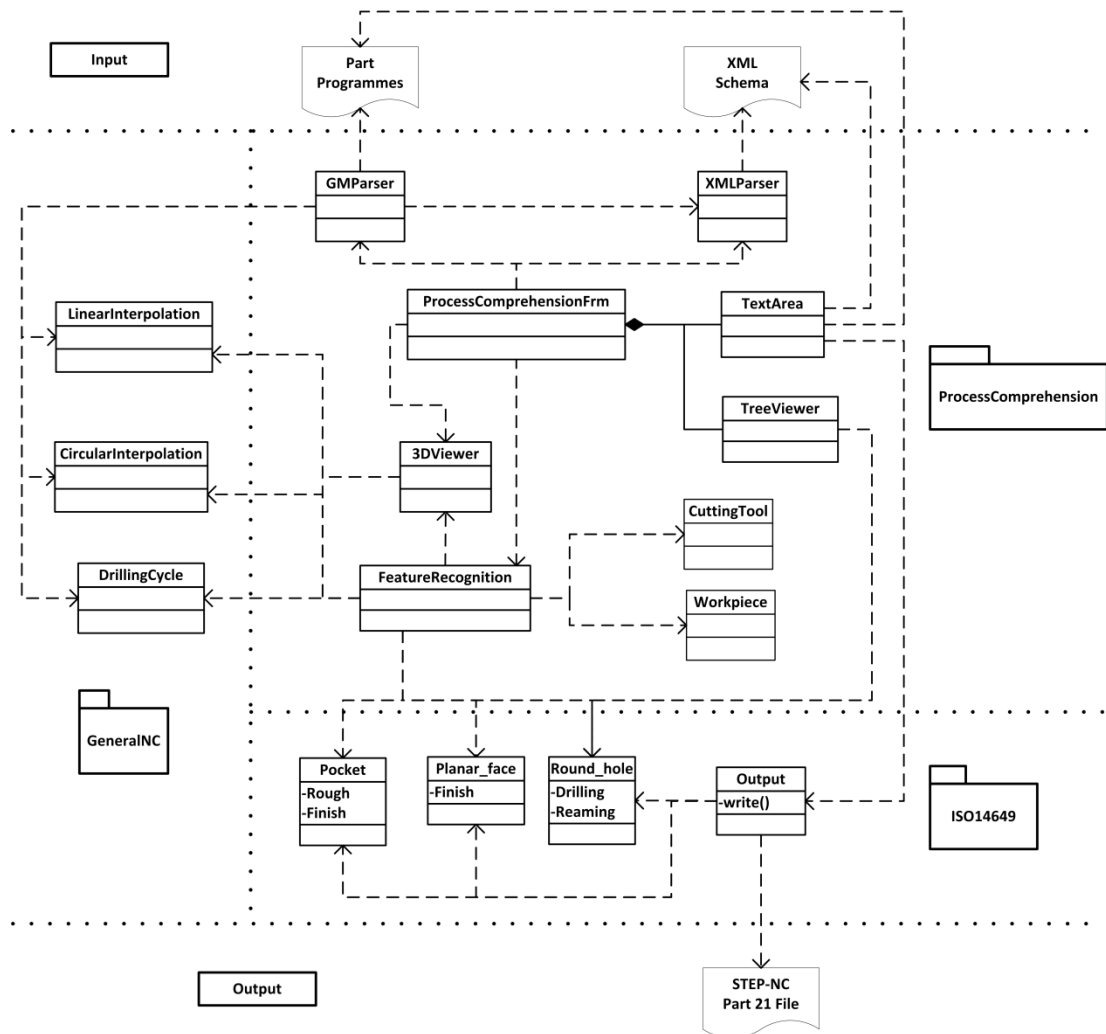


Figure 8.15 - Implementation overview of UPCi

8.9. Summary

In this chapter a prototype implementation of UPCi has been realised. It has the basic functions to translate the part programmes into a Meta-model and using this Meta data to reconstruct the original process plan including the part geometries. The prototype can be used to test the feasibility of process comprehension approach for prismatic part manufacturing.

9. Evaluation of UPCi prototype

9.1. Introduction

In order to evaluate the applicability of UPCi three example parts with common prismatic features have been utilised to illustrate the interoperable manufacturing scenario within the limitations of the prototype. The example parts have been designed in CAD/CAM system and post-processed into CNC part programmes. The part programmes have been used as the input into UPCi and STEP-NC representations were generated. Two types CNC part programme have been generated to test the capability of UPCi to support different programming dialects. Each part will be machined on two different CNC machines. One was machined on a Fanuc machine and the other one was machined on another machine with a Siemens controller. The part from the Fanuc machine is machined from the part programmes post-processed from the CAM system and updated with cutting parameter changes at the machine. The part programme used on Siemens machine was generated from the corresponding STEP-NC file which is the output of UPCi comprehended from the Fanuc part programme. The parts machined on the two machines were used to validate the similarities between the original part programmes and the STEP-NC representations. The validation results and analysis have been presented.

9.2. Evaluation method

To evaluate the UPCi, two set of tests of three test parts have been designed for different purposes. The first test (Test I) is to validate the capability of UPCi to deal with different programming dialects. The method is to compare two STEP-NC files generated from two different NC part programmes for the same component and check the semantic identities. The NC part programmes have been generated from the same CAD/CAM system using different post processors for two controller dialects (Fanuc and Siemens). The flow chart of Test I is shown in Figure 9.1.

To validate the Meta-model method for reading in different programming dialects through a standard interface, the expected results of Test I are: two STEP-NC files generated from different CNC part programmes of the same part should be semantically identical.

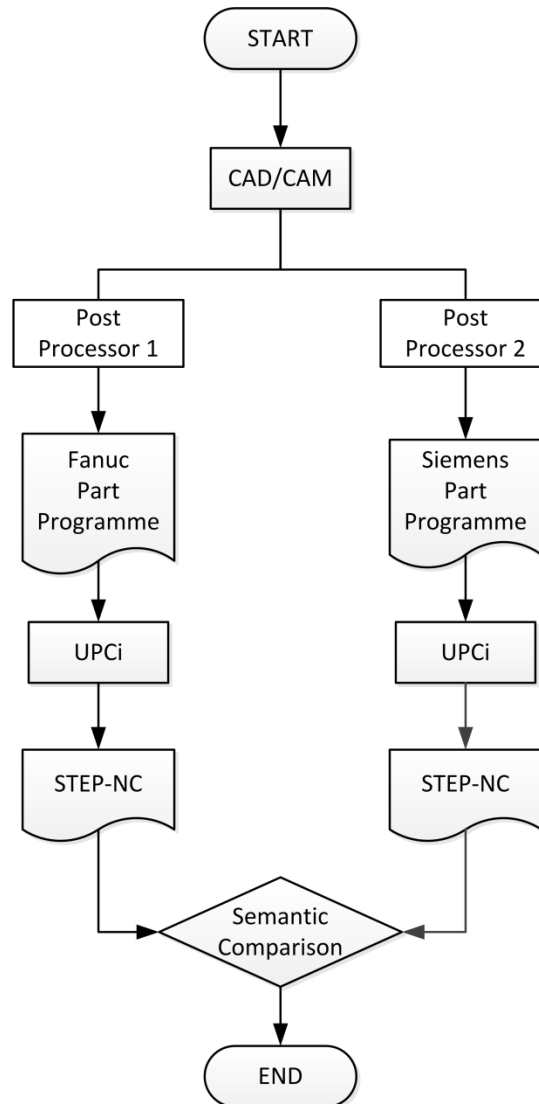


Figure 9.1 - Evaluation method: Test I

To evaluate the efficiency of the approach of process comprehension and display the application of knowledge reuse through process comprehension, a further verification is needed to evaluate: (i) feature recognition algorithm; (ii) the consistency between the STEP-NC file and the original process plan in the original part programme. Hence, Test II is proposed as illustrated in Figure 9.2.

The method of Test II is to compare two machined parts: one is from the traditional CAD/CAM/CNC manufacturing process and the other one is based on the STEP-NC process plan. The first stage is to design and generate the NC part programme for a part, then machine it on one machine. The next stage is to input the part programme into UPCi to get the STEP-NC process plan. Following this the STEP-NC process plan is used to generate part programme for another machine and produce the part. Finally a

comparison of the two parts is conducted to verify semantic consistency between the STEP-NC representation and the part programme. To prove the process consistent between the STEP-NC representation and the NC part programme, the two machined parts should be geometrically identical.

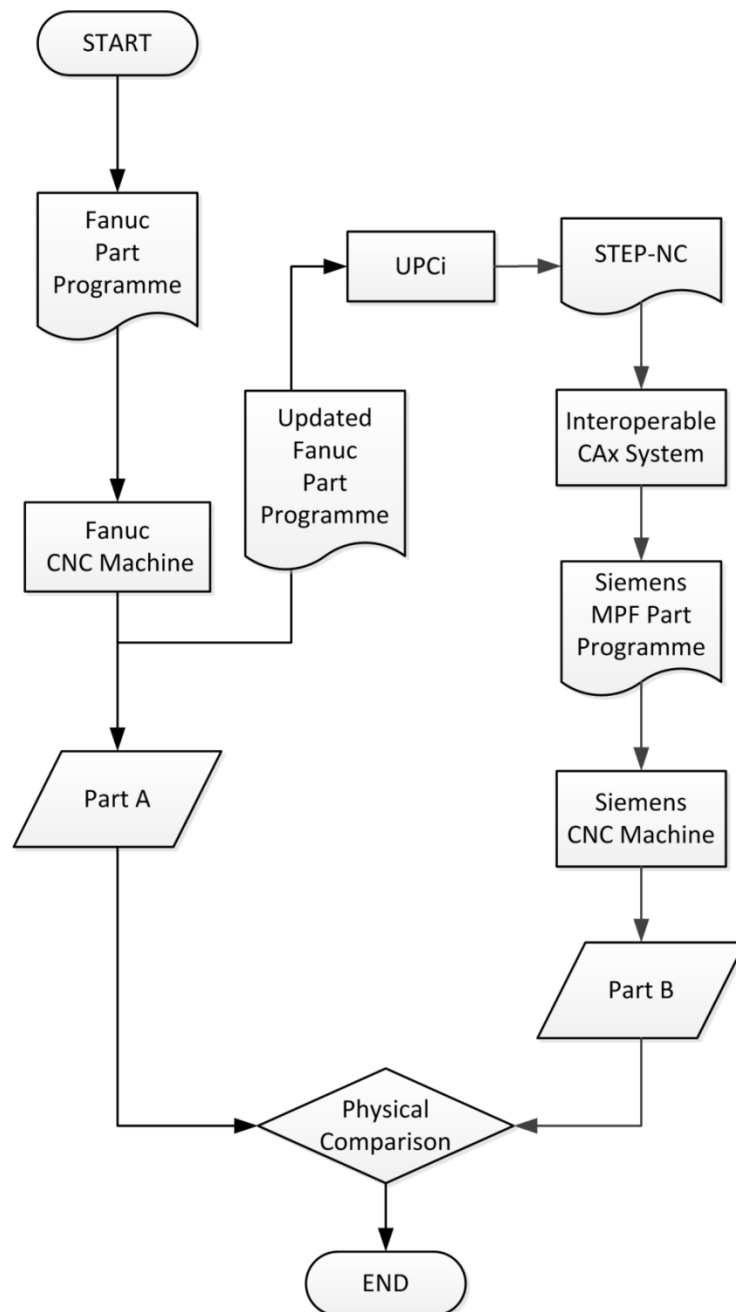


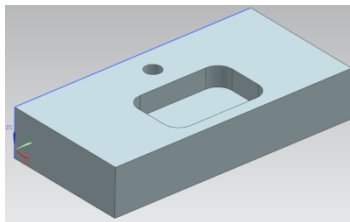
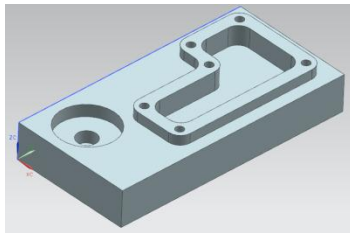
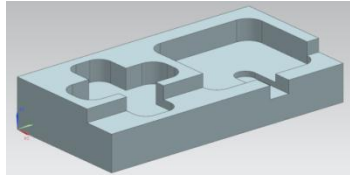
Figure 9.2 - Evaluation method: Test II

To carry out the tests, three test parts have been designed which range from parts consisting autonomous features to parts with one or more feature interactions. Six of the most common prismatic features to be tested are as follows:

- Planar face
- Hole
- Pocket
- Slot
- Step
- Boss

Table 9.1 outlines each test part and the evaluation criteria that it has been designed to test.

Table 9.1 - Test parts

	Component	Evaluation criteria
Part I		<ul style="list-style-type: none"> - Recognition of autonomous features: a planar face, a rectangular pocket and a hole. - Rough/Finish operation identification. - Process comprehension. - Knowledge reuse.
Part II		<ul style="list-style-type: none"> - Recognition of irregular-shaped interacting features: a pocket with in a boss and a round hole inside of a circular pocket. - Rough/Finish operation identification. - Process comprehension. - Knowledge reuse.
Part III		<ul style="list-style-type: none"> - Recognition of interacting features: three overlapped pockets, a pocket interacting with a step and a slot. - Rough/Finish operation identification. - Process comprehension. - Knowledge reuse.

9.3. Test I

Three test parts below have been used to illustrate the evaluation method: Test I as shown in Figure 9.1.

9.3.1. Part I

The first part, adapted from ISO 14649-11, has been designed in a CAD/CAM system with three features: a planar face, a pocket and a round hole. This part is selected in order to take advantage of the standardised STEP-NC file in ISO 14649-11. The generated STEP-NC files from UPCi can be utilised to compare with the standard file to evaluate the UPCi implementation. The comparison method used in this research is to compare key entities in the Data section of the STEP-NC files. The Header section is out of the comparison since there is no process information there.

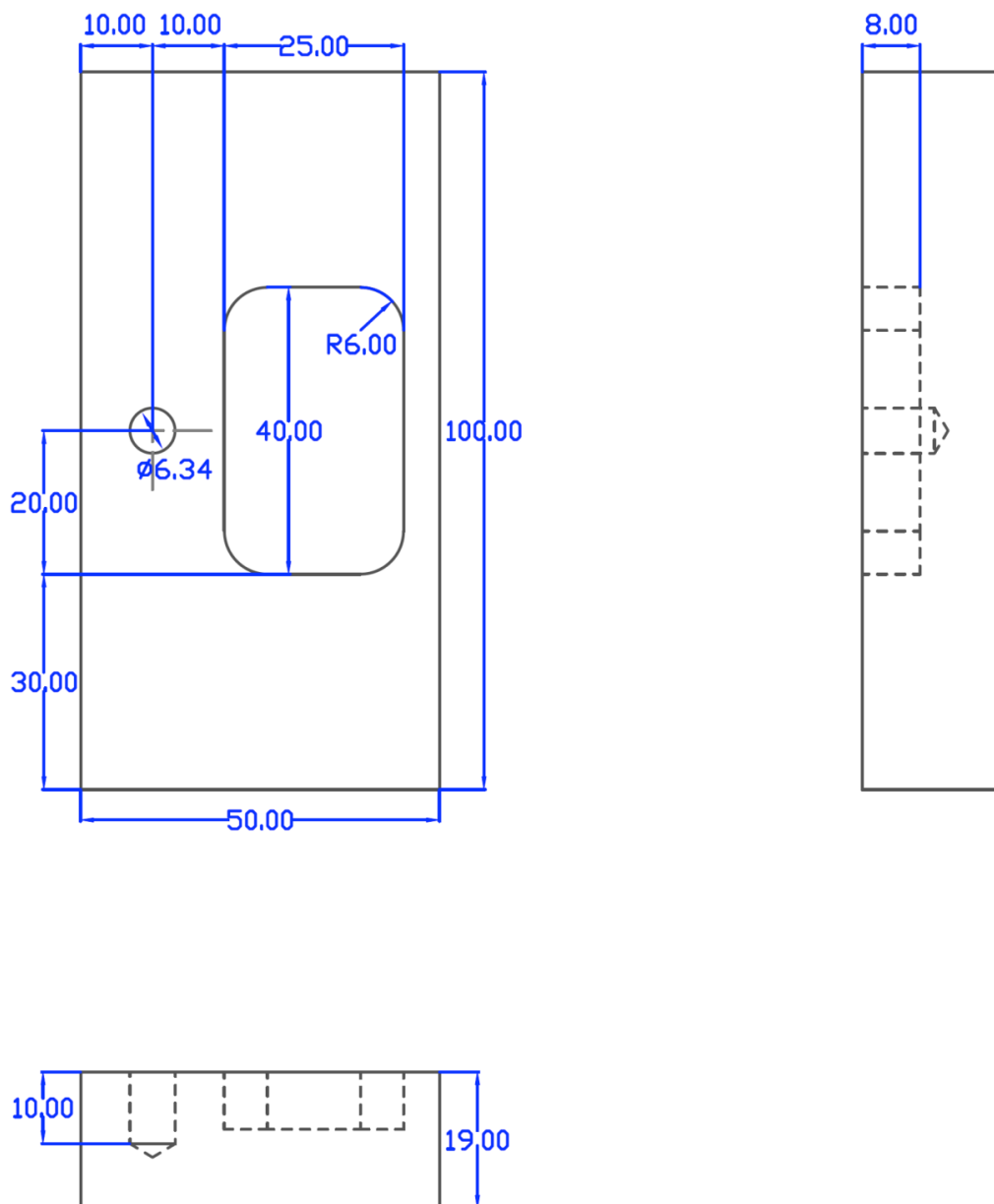


Figure 9.3 - Part I drawing

The part is to be cut from a 50mm x 100mm x 20mm block. The features and the operation that makes up the process plan are as follows in Table 9.2:

In this test, the part has been post-processed from CAM system for two CNC machines: a four-axis CNC machine equipped with a Fanuc 18i controller and a four-axis machine with Siemens 840D. The part programmes can be found in the Appendix C.1. Two STEP-NC presentations of the process plan have been generated (Appendix C.2). The comparison between the two STEP-NC files can be used to evaluate the capability of UPCi to comprehend process information from different programming languages. To compare them with the standardised STEP-NC code in ISO 14649-11, the consistency between them is used as the benchmark to evaluate the integrity of the translation from part programme to STEP-NC process plan. The comparison result is given in Table 9.3 and Table 9.4.

The two generated STEP-NC files from UPCi are almost identical. They have the same number of entities and the entities are in the same order. The results are listed in Table 9.3.

Table 9.2 - Process plan of Part I

Features	Operations	Tools
Planar face	Finish planar face	FaceMill Ø66
Round hole	Drill hole	Drill Ø6.2
	Ream hole	Reamer Ø6.6
Pocket	Rough pocket	SlotDrill Ø12
	Finish pocket	SlotDrill Ø12

Table 9.3 - Comparison between two generated STEP-NC files for Part I

1	Fanuc	#26=MILLING_CUTTING_TOOL('T8',#32,(),\$,,\$,\$);
	Siemens	#26=MILLING_CUTTING_TOOL('T1',#32,(),\$,,\$,\$);
2	Fanuc	#34=CARTESIAN_POINT('PLANAR_FACE1',(90.0,20.0,0.0));
	Siemens	#34=CARTESIAN_POINT('PLANAR_FACE1',(83.0,20.0,0.0));
3	Fanuc	#41=TOLERANCED_LENGTH_MEASURE(189.0,\$);
	Siemens	#41=TOLERANCED_LENGTH_MEASURE(182.0,\$);
4	Fanuc	#50=MILLING_CUTTING_TOOL('T9',#54,(),\$,,\$,\$);
	Siemens	#50=MILLING_CUTTING_TOOL('T2',#54,(),\$,,\$,\$);
5	Fanuc	#56=MILLING_CUTTING_TOOL('T7',#60,(),\$,,\$,\$);
	Siemens	#56=MILLING_CUTTING_TOOL('T3',#60,(),\$,,\$,\$);
6	Fanuc	#77=MILLING_CUTTING_TOOL('T10',#83,(),\$,,\$,\$);
	Siemens	#77=MILLING_CUTTING_TOOL('T4',#83,(),\$,,\$,\$);

Between these two STEP-NC files there are 6 differences, 4 of which are tool names. It is common that tools with the same dimension have different names on two different machines. The other two differences (number 2 and 3) are due to the different start points to mill the top surface. Actually, the difference starts from shopfloor changes to the Fanuc part programme, listed in Appendix C. At the shopfloor it was thought it is a good practice to place the tool a little far from the part and then cut into the material for safety reasons. However, it doesn't make a difference to the feature and the machining method. Hence, the two STEP-NC file generated from Fanuc and Siemens part programmes are semantic identical.

In Table 9.4, the comparison listed is between the standard STEP-NC file from ISO 14649-11 and the one generated by UPCi from Fanuc part programme.

Table 9.4 - Comparison between the standard file and the generated file

Entities	ISO14649-11	127 entities
	Fanuc	158 entities
Workingsteps	ISO14649-11	#10=MACHINING_WORKINGSTEP('WS FINISH PLANAR FACE1',#62,#16,#19,\$); #11=MACHINING_WORKINGSTEP('WS DRILL HOLE1',#62,#17,#20,\$);

		#12=MACHINING_WORKINGSTEP('WS REAM HOLE1',#62,#17,#21,\$); #13=MACHINING_WORKINGSTEP('WS ROUGH POCKET1',#62,#18,#22,\$); #14= MACHINING_WORKINGSTEP('WS FINISH POCKET1',#62,#18,#23,\$);
	Fanuc	#4=MACHINING_WORKINGSTEP('WS PLANAR_FACE1',#10,#11,#12,\$); #5=MACHINING_WORKINGSTEP('WS DRILLING ROUND HOLE 6.34',#10,#44,#45,\$); #6=MACHINING_WORKINGSTEP('WS REAMING ROUND HOLE 6.34',#10,#44,#46,\$); #7=MACHINING_WORKINGSTEP('WS ROUGH POCKET2',#10,#69,#70,\$); #8=MACHINING_WORKINGSTEP('WS FINISH POCKET2',#10,#69,#71,\$);
Features	ISO1464 9-11	#16=PLANAR_FACE('PLANAR_FACE1',#4,(#19),#77,#63,#24,#25,\$,()); #17=ROUND_HOLE('HOLE1 D=22MM',#4,(#20,#21),#81,#64,#58,\$,#26); #18=CLOSED_POCKET('POCKET1',#4,(#22,#23),#84,#65,\$,#27,#35,#37,#28);
	Fanuc	#11=PLANAR_FACE('PLANAR_FACE1',#3,(#12),#17,#18,#19,#20,\$,()); #44=ROUND_HOLE('ROUND HOLE 6.34',#3,(#45,#46),#47,#48,#49,\$,\$); #69=CLOSED_POCKET('POCKET2',#3,(#70,#71),#72,#73,\$,\$,#74,#75,\$,#76);
Operations	ISO1464 9-11	#19=PLANE_FINISH_MILLING(\$,\$,'FINISH PLANAR_FACE1',10.000,\$,#39,#40,#41,\$,#60,#61,#42,2.500,\$); #20=DRILLING(\$,\$,'DRILL HOLE1',10.000,\$,#44,#45,#41,\$,\$,\$,\$,#46); #21=REAMING(\$,\$,'REAM HOLE1',10.000,\$,#47,#48,#41,\$,\$,\$,\$,#49,T,\$,\$); #22=BOTTOM_AND_SIDE_ROUGH_MILLING(\$,\$,'ROUGH POCKET1',15.000,\$,#39,#50,#41,\$,\$,\$,#51,2.500,5.000,1.000,0.500); #23=BOTTOM_AND_SIDE_FINISH_MILLING(\$,\$,'FINISH POCKET1',15.000,\$,#39,#52,#41,\$,\$,\$,#53,2.000,10.000,\$,\$);
	Fanuc	#12=PLANE_ROUGH_MILLING(\$,\$,'PLANAR_FACE1',,\$,\$,#26,#27,#28,\$,#29,#30,#31,\$,\$); #45=MULTISTEP_DRILLING(\$,\$,'DRILLING ROUND HOLE 6.34',,\$,\$,#50,#51,#52,\$,\$,\$,\$,#53,0.0,6.3,6.3,\$); #46=REAMING(\$,\$,'REAMING ROUND HOLE 6.34',,\$,\$,#56,#57,#58,\$,\$,\$,\$,#59,F,\$,\$); #70=BOTTOM_AND_SIDE_ROUGH_MILLING(\$,\$,'ROUGH POCKET2',,\$,\$,#77,#78,#79,\$,#80,#81,#82,\$,\$,0.5,1.0); #71=BOTTOM_AND_SIDE_FINISH_MILLING(\$,\$,'FINISH POCKET2',,\$,\$,#77,#85,#86,\$,#87,#88,#89,\$,\$,0.0,0.0);

From the comparison result in the table above, the generated STEP-NC file has the same entities with the original file in the ISO 14649-11. They have same features and their operations in the same order of organised into workingsteps. The differences are entities numbers, feature geometry data (the test part is smaller), and several new entities in the generated file. They can be explained as follows:

The numbers of lines (entities, each line is a STEP-NC entity) are different since the generated file has more information than the standard one. In the standard file, there are no approach/retract strategies. In the lines with the index of #22 and #23, the approach /retract strategies for the rough and finish operation of the pocket is left as \$. They are optional according to the standard. However, from the part programmes, the approach and retract strategies can be recognised. They do exist in the part programme. Another difference is the pocket geometry data. In this research, the test part is smaller than the one in STEP-NC standard and the geometry data is expressed in a different methods or sets of entities. In the original file, the pocket shape is described as a closed pocket with the shape of a rectangular_closed_profile. To define the shape, the placement, width and length are needed. In the generated file, the pocket shape is described as a general_closed_profile, which has eight segments of arcs and poly lines to form the shape. This is the second reason why the generated file is longer than the one in the standard. Another reason why the differences exist between these two files is the cutting tool information. In the generated file, beside the tool type, there are only tool diameter and tool tip radius data. This is due to the purpose of simplifying the input of the UPCi. It is not practical and unnecessary as well to ask the user to specify the detailed tool information. In the standard file, apart from that, the tool body dimension is also included.

Basically the integrity of the generated STEP-NC file is satisfactory. It covers all the necessary entities to describe the part and its machining method. Additionally, it contains entities describing the machining strategies compared with the file from the standard.

9.3.2. Part II

Part II has been designed with the intention to prove the ability of UPCi to process the irregular-shaped features and interacting features. In this part, there is one circular pocket with a round hole, a non-rectangular boss with a pocket and a pattern of six deep holes. These features are to be cut from a 50mm x 100mm x 20mm block. The drawing of the part is provided in Figure 9.4.

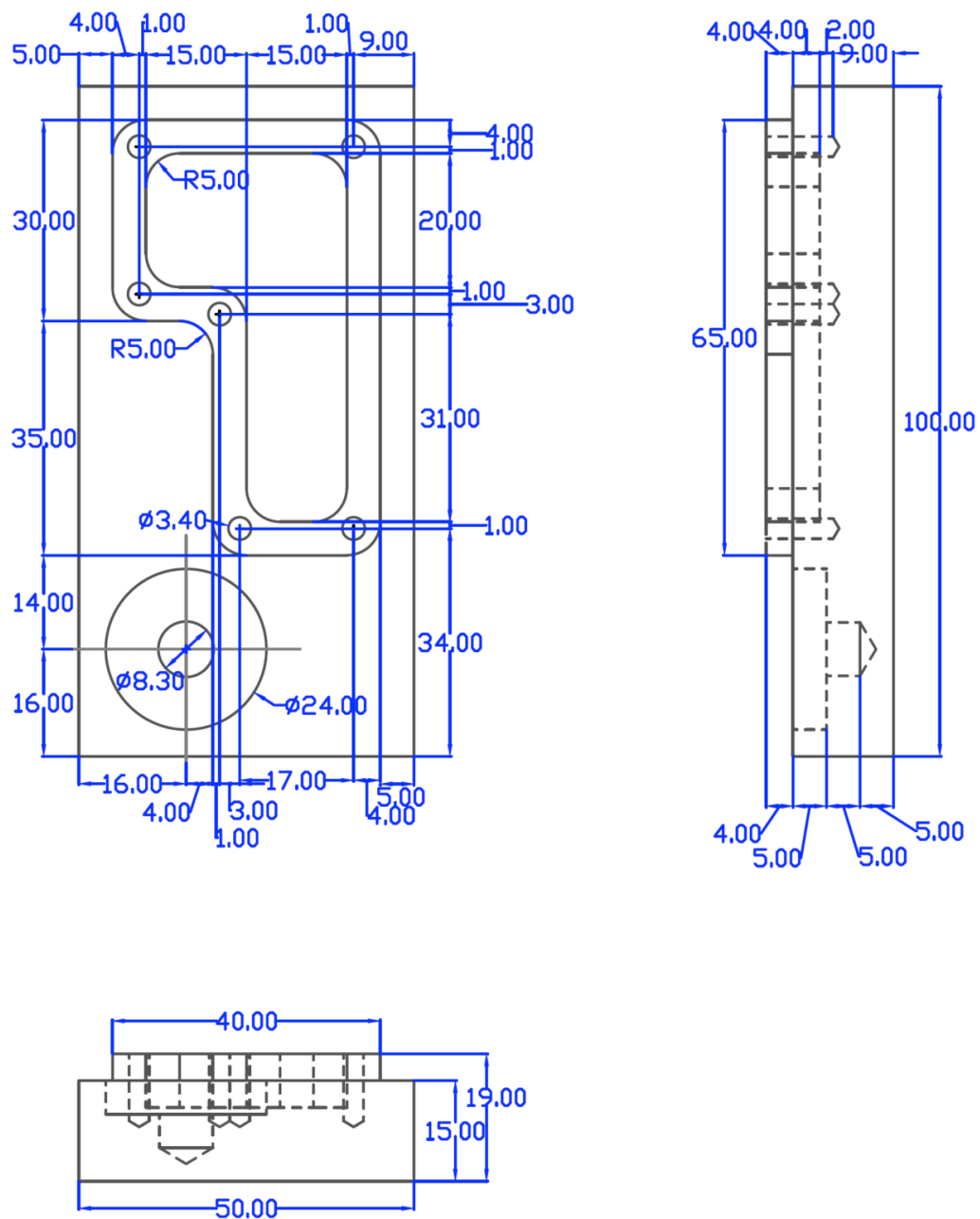


Figure 9.4 - Part II drawing

The features and the operation that makes up the process plan are as follows in Table 9.5. The part programmes post processed for Fanuc and Siemens controllers can be found in the Appendix C.1. The corresponding STEP-NC files process comprehended from these two part programmes are almost identical except the start point of the face milling operation and tool names. The STEP-NC files can be found in Appendix C.2. The comparison of the two STEP-NC files is listed in Table 9.6 below.

Table 9.5 - Process plan of Part II

Features	Operations	Tools
Planar face	Finish Planar face	FaceMill Ø66
Boss	Rough boss	SlotDrill Ø10
	Finish boss	SlotDrill Ø10
Pocket	Rough pocket	SlotDrill Ø8
	Finish pocket	SlotDrill Ø8
Circular pocket	Rough circular pocket	SlotDrill Ø8
Round hole	Drill hole	Drill Ø8.3
Small round holes	Drill holes	Drill Ø3.4

Table 9.6 - Comparison between two generated STEP-NC files for Part II

1	Fanuc	#34=MILLING_CUTTING_TOOL('T8',#40(),\$,,\$,\$);
	Siemens	#34=MILLING_CUTTING_TOOL('T1',#40(),\$,,\$,\$);
2	Fanuc	#42=CARTESIAN_POINT('PLANAR_FACE1',(90.0,20.0,0.0));
	Siemens	#42=CARTESIAN_POINT('PLANAR_FACE1',(83.0,20.0,0.0));
3	Fanuc	#49=TOLERANCED_LENGTH_MEASURE(189.0,\$);
	Siemens	#49=TOLERANCED_LENGTH_MEASURE(182.0,\$);
4	Fanuc	#60=MILLING_CUTTING_TOOL('T5',#66(),\$,,\$,\$);
	Siemens	#60=MILLING_CUTTING_TOOL('T2',#66(),\$,,\$,\$);
5	Fanuc	#176=MILLING_CUTTING_TOOL('T4',#182(),\$,,\$,\$);
	Siemens	#176=MILLING_CUTTING_TOOL('T3',#182(),\$,,\$,\$);
6	Fanuc	#304=MILLING_CUTTING_TOOL('T7',#308(),\$,,\$,\$);
	Siemens	#304=MILLING_CUTTING_TOOL('T4',#308(),\$,,\$,\$);
7	Fanuc	#322=MILLING_CUTTING_TOOL('T22',#326(),\$,,\$,\$);
	Siemens	#322=MILLING_CUTTING_TOOL('T5',#326(),\$,,\$,\$);

9.3.3. Part III

Part III has been designed to evaluate the capability of UPCi to process the interacting features. In this part, there are 4 circular pockets, a slot and a step with interactions. These features are to be cut from a 50mm x 100mm x 20mm block. The drawing of the part is illustrated in Figure 9.5.

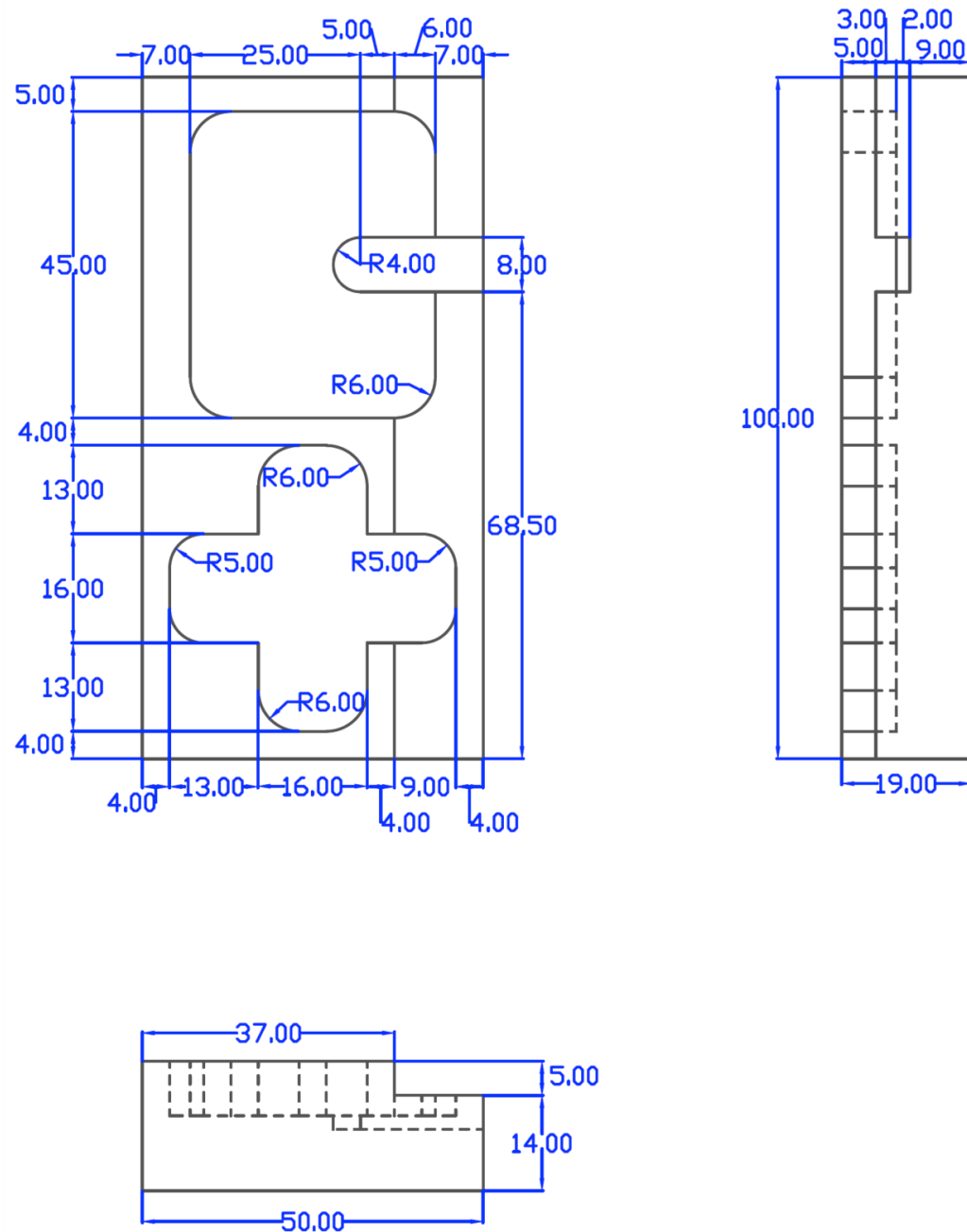


Figure 9.5 - Part III drawing

The features in this part are interacting with each other as shown in Figure 9.6:

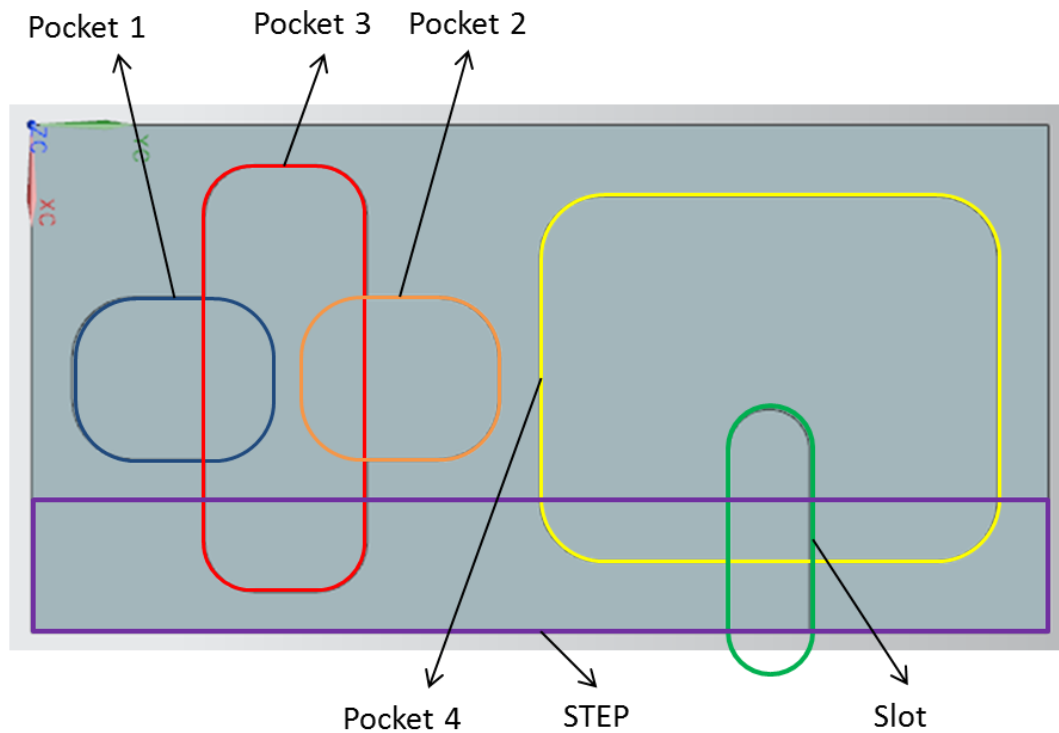


Figure 9.6 – Interacting features in Part III

The features and the operations that make up the process plan are as follows in Table 9.7:

Table 9.7 - Process plan of Part III

Features	Operations	Tools
Planar face	Finish planar face	FaceMill Ø66
Pocket 1	Rough pocket 1	SlotDrill Ø12
Pocket 2	Rough pocket 2	SlotDrill Ø12
Pocket 3	Rough pocket 3	SlotDrill Ø10
	Finish pocket 3	SlotDrill Ø10
Pocket 4	Rough pocket 4	SlotDrill Ø10
	Finish pocket 4	SlotDrill Ø8
Step	Rough step	SlotDrill Ø16
Slot	Rough slot	SlotDrill Ø8

The part programmes post processed for Fanuc and Siemens controllers can be found in the Appendix C.1. Similarly as Part I and Part II, the corresponding STEP-NC files process comprehended from these two part programmes are almost identical except the start

point of the face milling operation and tool names. One difference compared with Part II is the spindle speed for the face milling operation. The spindle speed in the Fanuc part programme has been optimised manually by the operator at the shopfloor. The STEP-NC files can be found in Appendix C.2. The comparison results of the two STEP-NC files are listed in Table 9.8 below.

Table 9.8 - Comparison between two generated STEP-NC files for Part III

1	Fanuc	#30=MILLING_CUTTING_TOOL('T8',#36(),\$,,\$,,\$);
	Siemens	#30=MILLING_CUTTING_TOOL('T1',#36(),\$,,\$,,\$);
2	Fanuc	#31=MILLING_TECHNOLOGY(0.016666666666666666,TCP.,\$,133.33333333333334,\$,\$,\$,\$,,\$);
	Siemens	#31=MILLING_TECHNOLOGY(0.134416666666666666,TCP.,\$,133.33333333333334,\$,\$,\$,\$,,\$);
3	Fanuc	#38=CARTESIAN_POINT('PLANAR_FACE1',(90.0,20.0,0.0));
	Siemens	#38=CARTESIAN_POINT('PLANAR_FACE1',(83.0,20.0,0.0));
4	Fanuc	#45=TOLERANCED_LENGTH_MEASURE(189.0,\$);
	Siemens	#45=TOLERANCED_LENGTH_MEASURE(182.0,\$);
5	Fanuc	#55=MILLING_CUTTING_TOOL('T10',#61(),\$,,\$,,\$);
	Siemens	#55=MILLING_CUTTING_TOOL('T2',#61(),\$,,\$,,\$);
6	Fanuc	#203=MILLING_CUTTING_TOOL('T5',#209(),\$,,\$,,\$);
	Siemens	#203=MILLING_CUTTING_TOOL('T3',#209(),\$,,\$,,\$);
7	Fanuc	#363=MILLING_CUTTING_TOOL('T2',#369(),\$,,\$,,\$);
	Siemens	#363=MILLING_CUTTING_TOOL('T5',#369(),\$,,\$,,\$);

9.4. Test II

The three test parts in the previous section are again used to illustrate the evaluation method: Test II as shown in Figure 9.2.

9.4.1. Part I

Part I has been designed to test ability of the prototype system to recognise individual features, identify the roughing and finishing operations and generate STEP-NC representation of process plan. Initially, the part programme post processed from CAM system for the Fanuc machine has been used to machine the part, as shown in Figure 9.7.



Figure 9.7 - Part I machined on Fanuc machine

Using the updated part programme from the Fanuc machine, UPCi recognised the process information and stored it in a STEP-NC file. Figure 9.8 pictures UPCi in action: defining workpiece, specifying cutting tool, identifying feature boundary and recognising the planar face feature. In the figure, the recognition of the planar face has been captured. The cutting tool used for the planar face operation is a 66mm facemill. Using the tool diameter, the covering area of the facing operation can be calculated. Then the feature can be identified as discussed in Chapter 7. The last step for this feature is to specify the strategies used in the operation with the visual aids of 3D viewer displaying the toolpath of the operation. After feature recognition, the final result is shown in Figure 9.9, which displays the STEP-NC file and tree view of the file. Five workingsteps including four feature and five operations have been recognised.



Figure 9.8 - UPCi during process comprehension of Part I

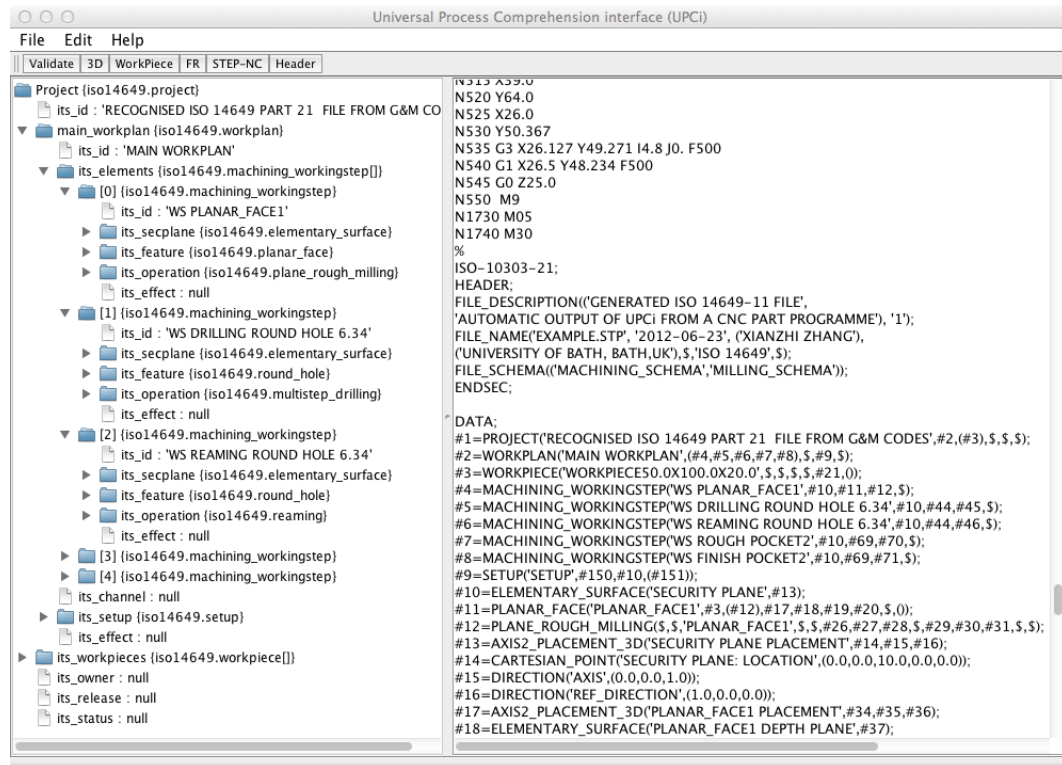


Figure 9.9 - Process comprehension results of Part I

Having generated the STEP-NC representation of the process plan, the STEP-NC file was then fed into an interoperable CAM system developed by Nassehi (Nassehi 2007). Figure 9.10 pictures the CAM system in action after reading in the STEP-NC file of Part I.

Through the CAM system, the STEP-NC presentation of the process plan has been post-processed into a MPF part programme for Siemens machine, as shown in Figure 9.11. MPF is a proprietary programming format for Siemens controllers. The file is provided in Appendix C.3. It is worth to mention that the MPF file in the appendix is generated from the STEP-NC file. To simulate or use it on the machine, there is a need to update the tool names. Since the MPF programme is feature based, it is possible to use a tool with a different diameter to machine the feature, which gives the operator flexibility at the shopfloor. This is another advantage achieved by interoperability in addition to the process knowledge reuse.

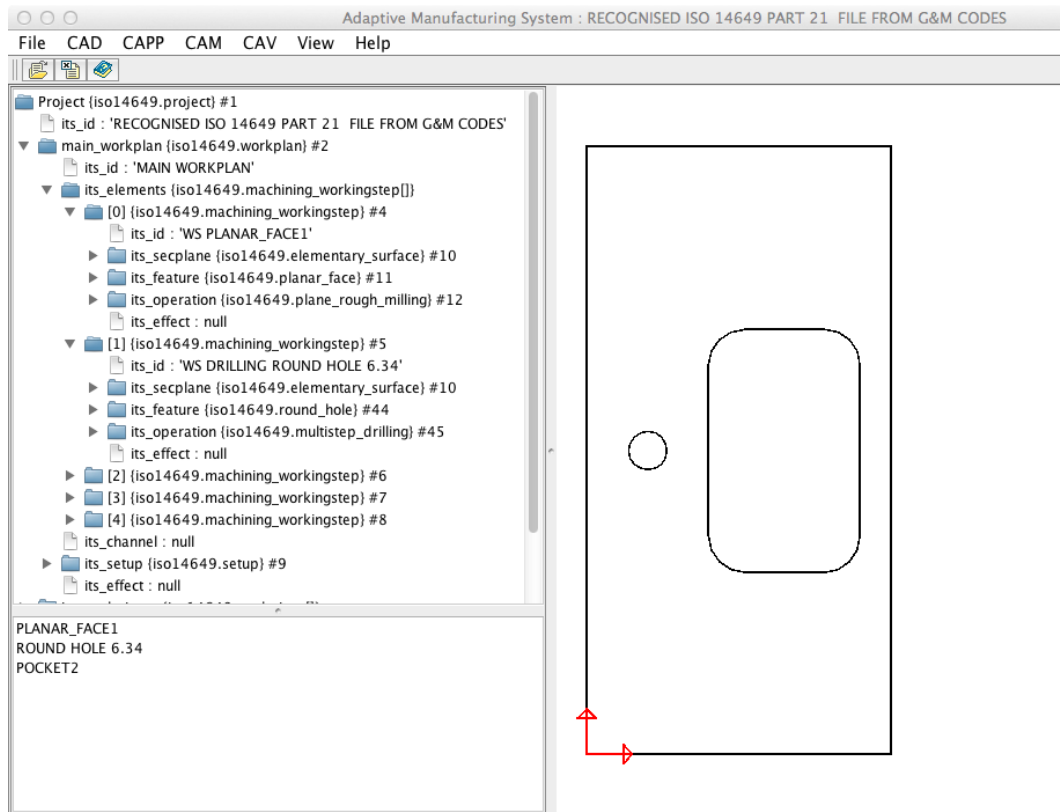


Figure 9.10 - Part I in the interoperable CAD/CAM system

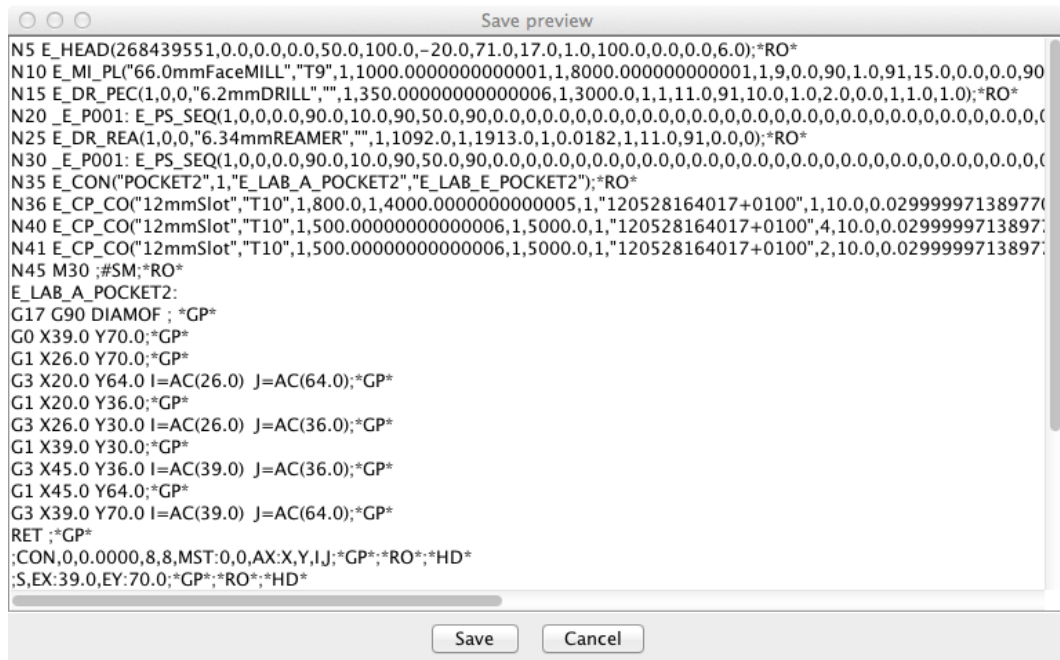


Figure 9.11 - Siemens MPF generated from STEP-NC file for Part I

The MPF file was then simulated in ShopMill (as shown in Figure 9.12), a shopfloor programming system from Siemens, and performed on the Siemens machine to produce another copy of Part I, as shown in Figure 9.13. The two machined parts, one from Fanuc

machine and another from Siemens machine, are shown in Figure 9.14. According to measurement, the two machined parts are geometrically identical.

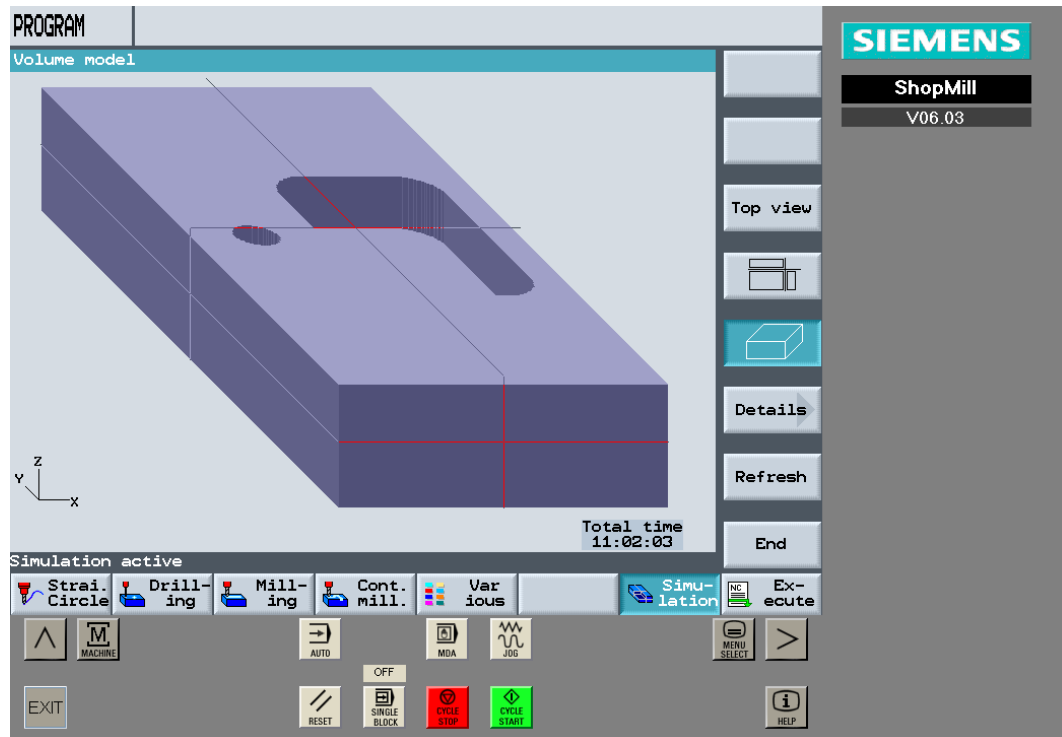


Figure 9.12 - Three dimensional simulation of Part I



Figure 9.13 - Part I machined on Siemens machine



Figure 9.14 - Test results for Part I

9.4.2. Part II

Part II has been designed to test the capability of UPCi prototype to process comprehend complex and interacting features. Utilising the part programme for the Fanuc machine in Test I. The part machined on the Fanuc machine is shown in Figure 9.15.

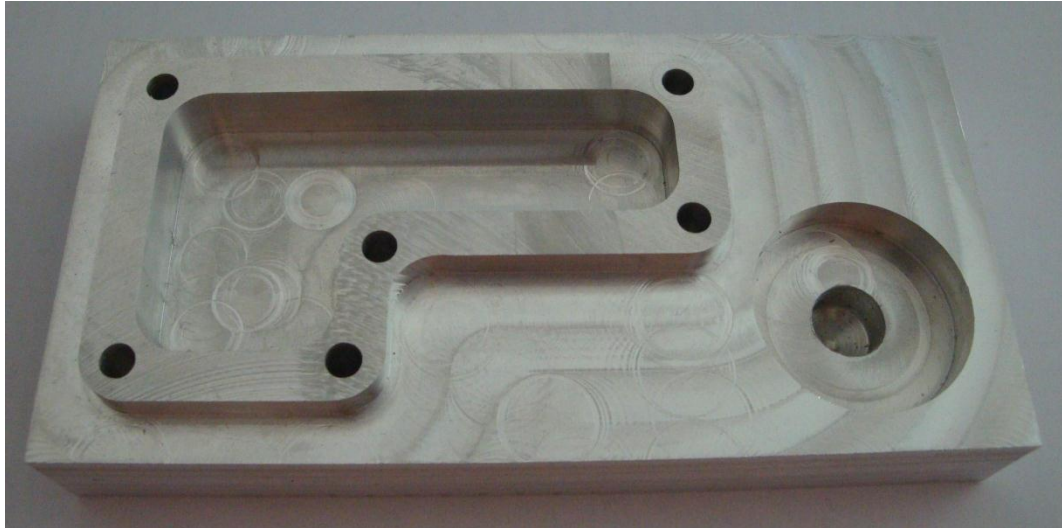


Figure 9.15 - Part II machined on Fanuc machine

The updated part programme from the Fanuc machine was then used as the input of UPCi prototype system to comprehend the process information. In Figure 9.16, the workpiece definition and the feature recognition of a planar face with a boss have been captured. The toolpath of the operation has been extracted and used to calculate the boundaries of the feature. After recognition of all the features, a STEP-NC representation of the process plan were generated. As shown in Figure 9.17, 11 features have been recognised with 13 operations organised in 13 workingsteps.

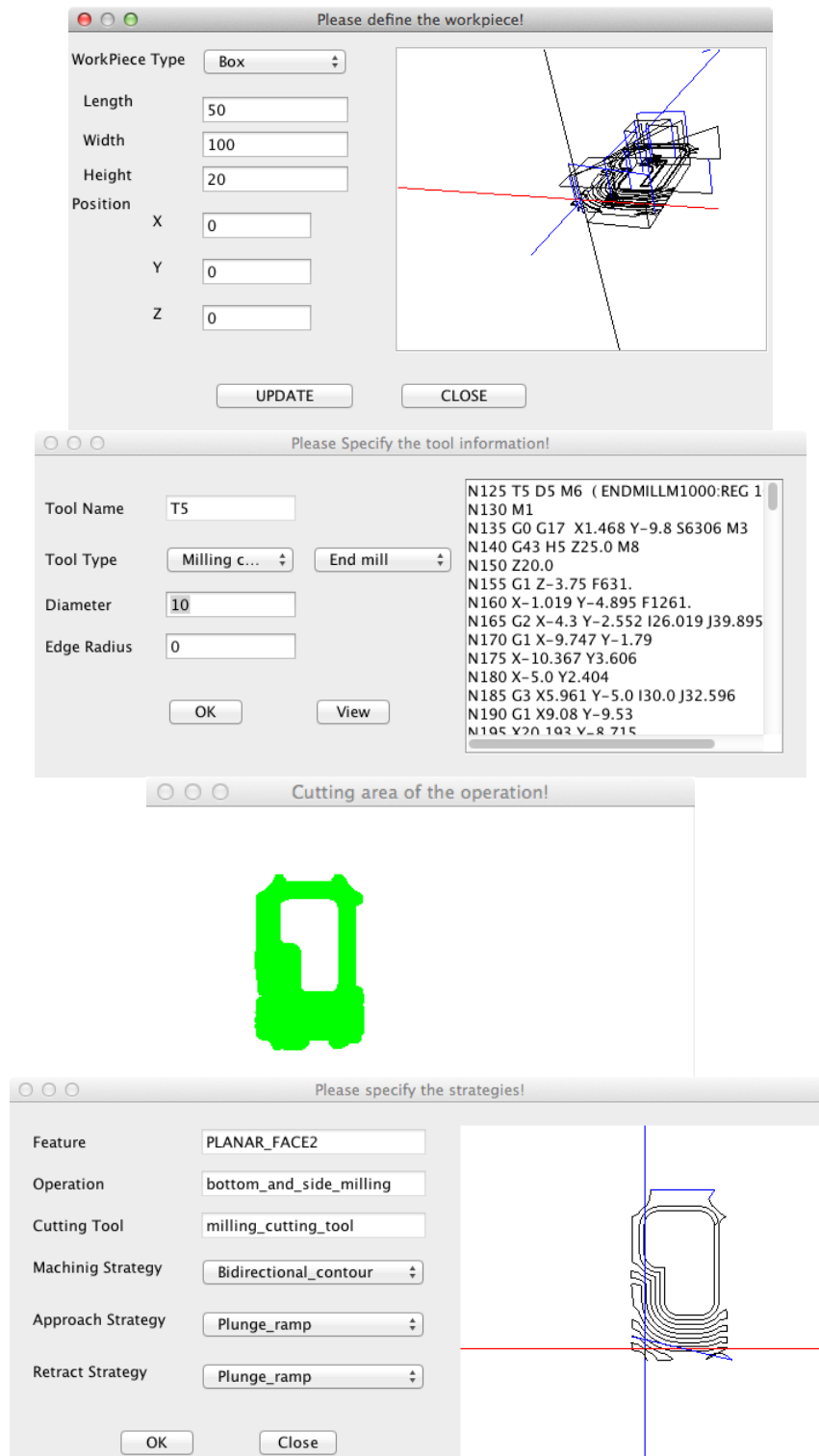


Figure 9.16 - UPCi during process comprehension of Part II

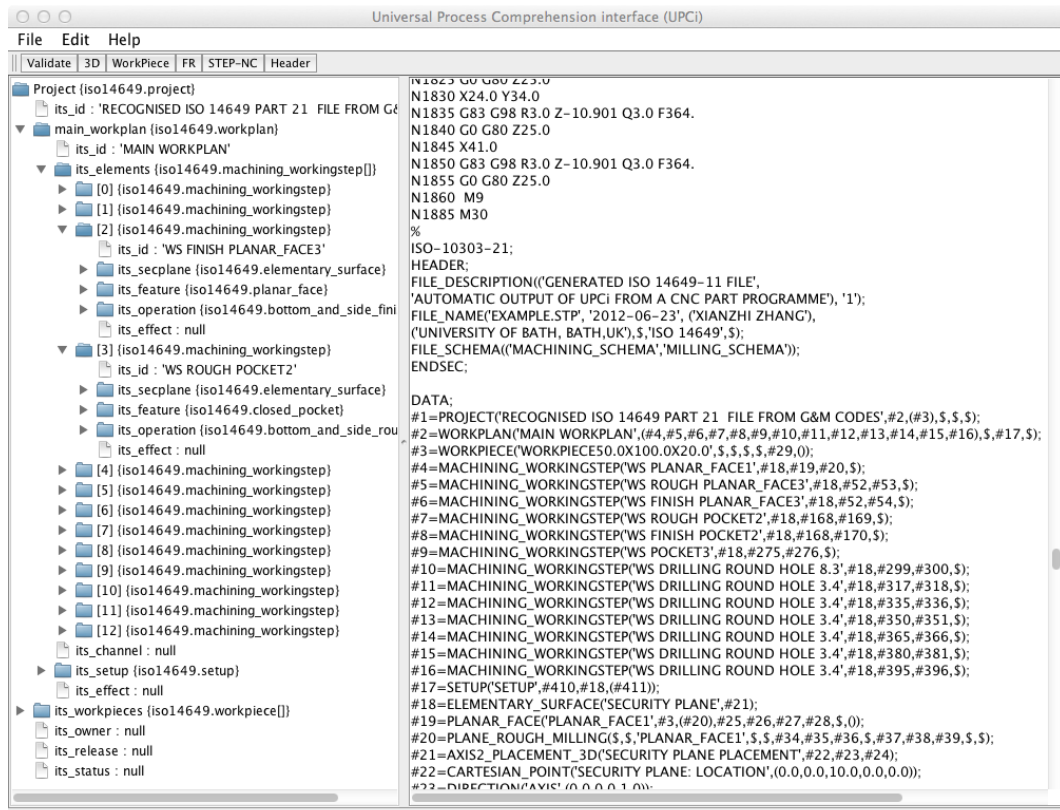


Figure 9.17 - Process comprehension results of Part II

Similar to Part I, the generated STEP-NC file was fed into the interoperable CAD/CAM system. Figure 9.18 pictures the CAD/CAM system in action with the Part II. Through the CAD/CAM system, the STEP-NC presentation of the process plan has been post-processed into a MPF part programme (see Appendix C.3) for the Siemens machine. The MPF file was then simulated in the ShopMill interface (as shown in Figure 9.19) before being applied on the Siemens machine.

The final part machined on the Siemens machine is shown in Figure 9.20. Together with the part machined on the Fanuc machine, the two machined parts, are shown in Figure 9.21. According to measurement, the two machined parts are geometrically identical.

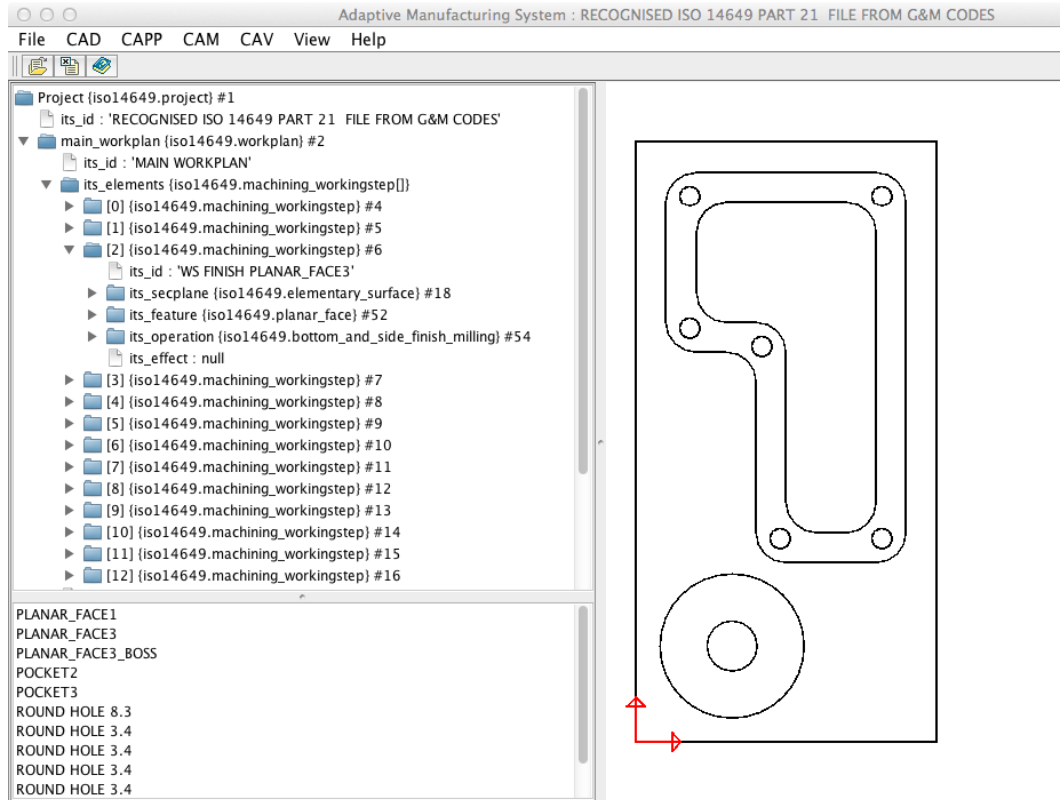


Figure 9.18 - Part II in the interoperable CAD/CAM system

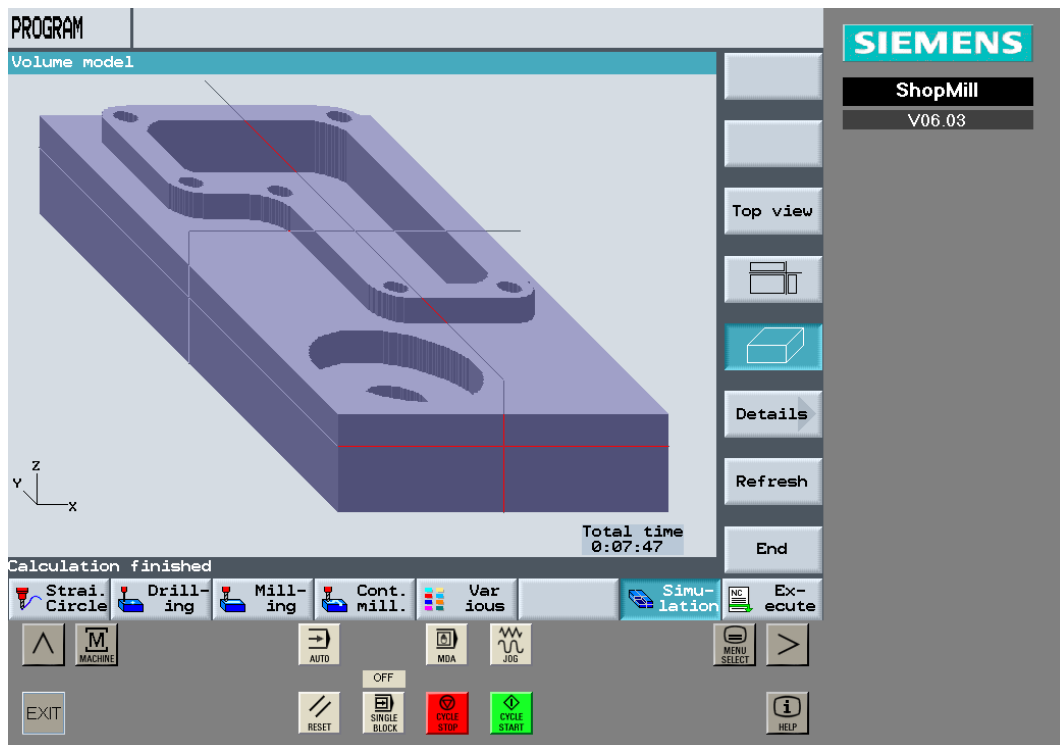


Figure 9.19 - Three dimensional simulation of Part II

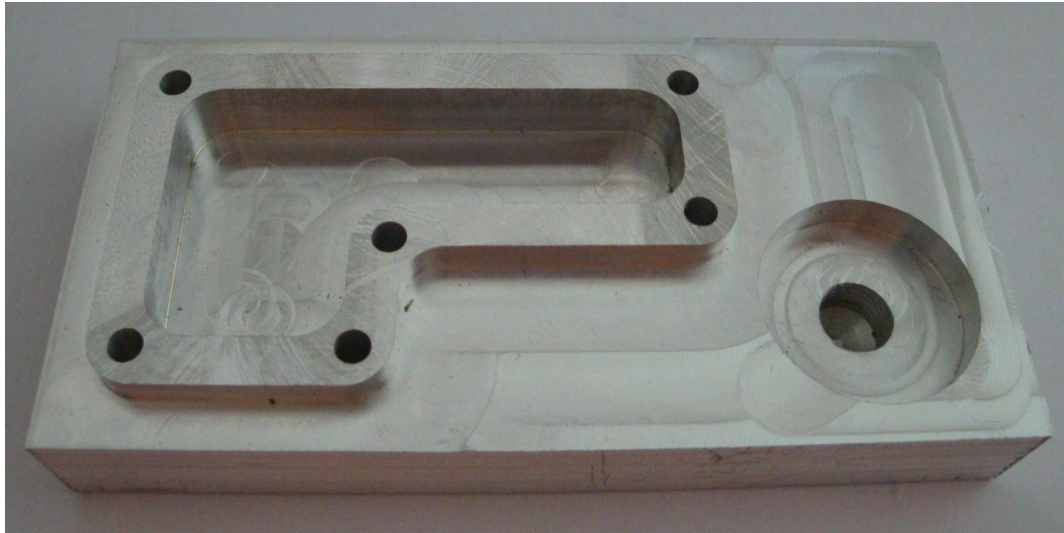


Figure 9.20 - Part II machined on the Siemens machine



Figure 9.21 - Test results for Part II

9.4.3. Part III

Part III has been designed with 4 pockets, a slot and a step interacting with each other to test the capability of the prototype system to recognise interacting features. Using the Fanuc part programme in Test I, the finished part is shown in Figure 9.22.



Figure 9.22 - Part III machined on Fanuc machine

As with the two previous parts, the updated part programme from the Fanuc machine was then used as the input of UPCI prototype system to comprehend the process information. In Figure 9.23, the workpiece definition and the feature recognition of a rectangular pocket have been captured. The toolpath of the operation has been extracted and used to calculate the boundaries of the feature. After recognition of all the features, a STEP-NC representation of the process plan were generated. As shown in Figure 9.24, 6 features have been recognised with 9 operations organised in 9 workingsteps.

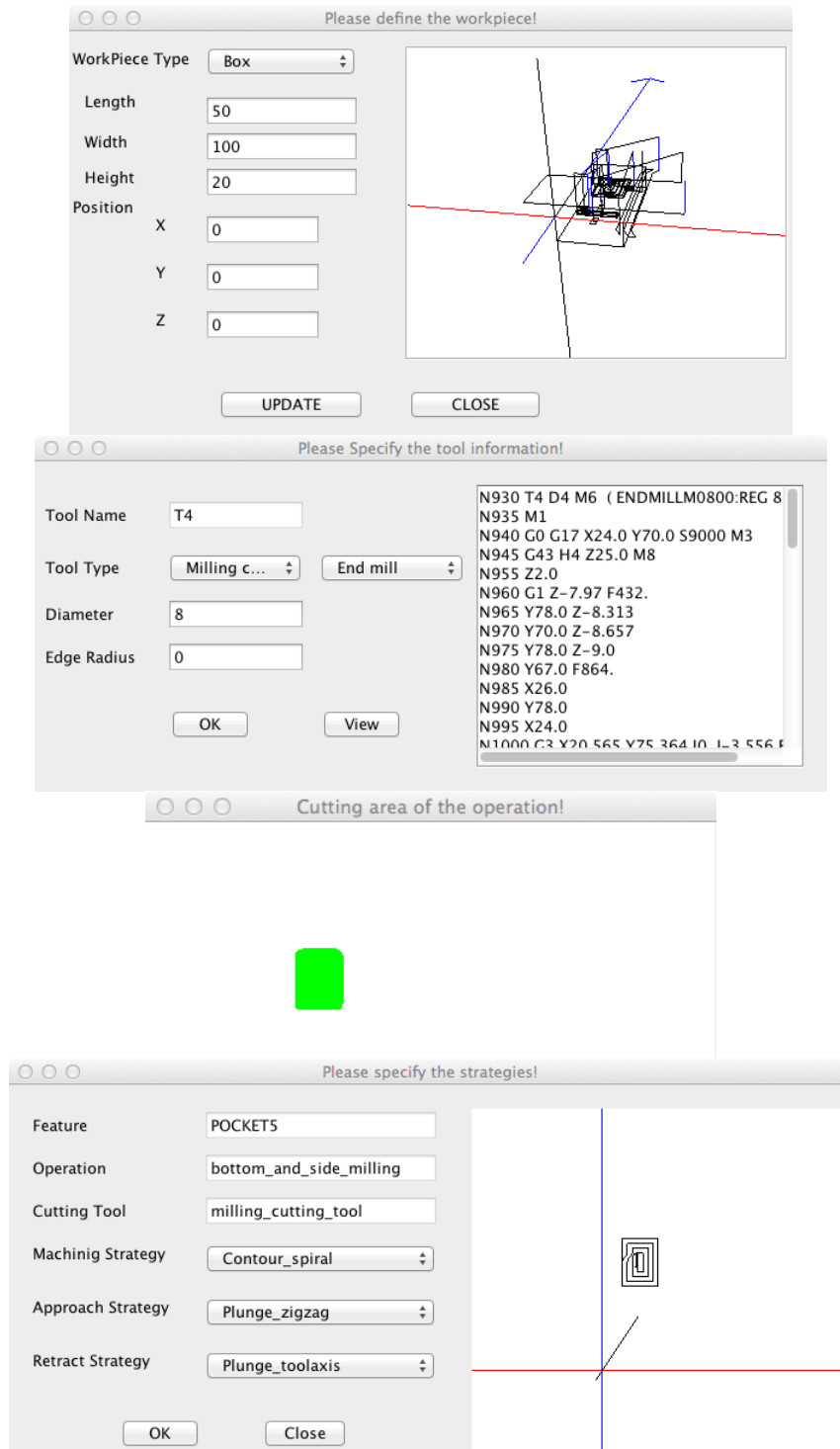


Figure 9.23 - UPCI during process comprehension of Part III

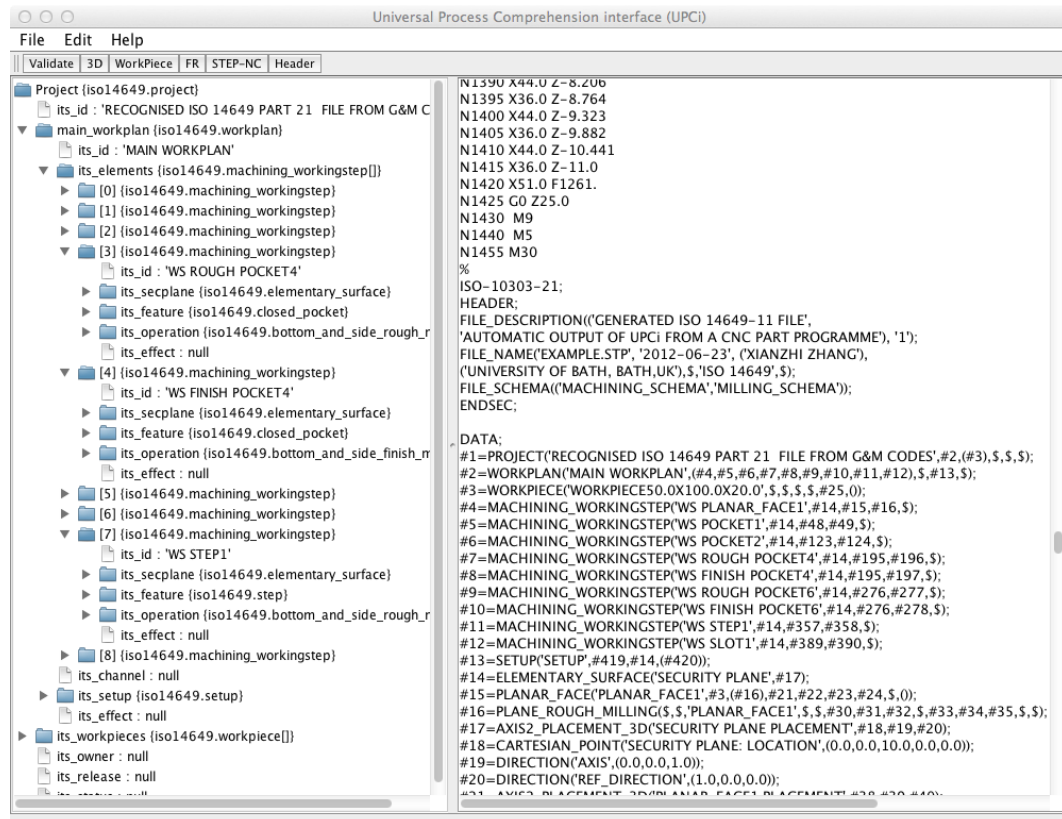


Figure 9.24 - Process comprehension results of Part III

The generated STEP-NC file was fed into the interoperable CAD/CAM system. Figure 9.25 pictures the CAD/CAM system in action with the Part III. Through the CAD/CAM system, the STEP-NC presentation of the process plan has been post-processed into a MPF part programme (see Appendix C.3) for Siemens machine. The MPF file was then simulated in ShopMill interface (as shown in Figure 9.26) before used on Siemens machine.

The final part machined on Siemens machine is shown in Figure 9.27. When machining this part, the $\varnothing 12$ SlotDrill was not available on the Siemens machine. A $\varnothing 10$ SlotDrill was used to machine the two small pockets. This is the flexibility advantage of interoperability as mentioned in Section 9.4.1. The workpiece for this part is also different from the previous two parts. A 50mm x 100mm x 50 block was used due to the unavailability of the 20mm thick block. Together with the part machined on Fanuc machine, the two machined parts, are shown in Figure 9.28. According to measurement, the two machined parts (features) are geometrically identical.

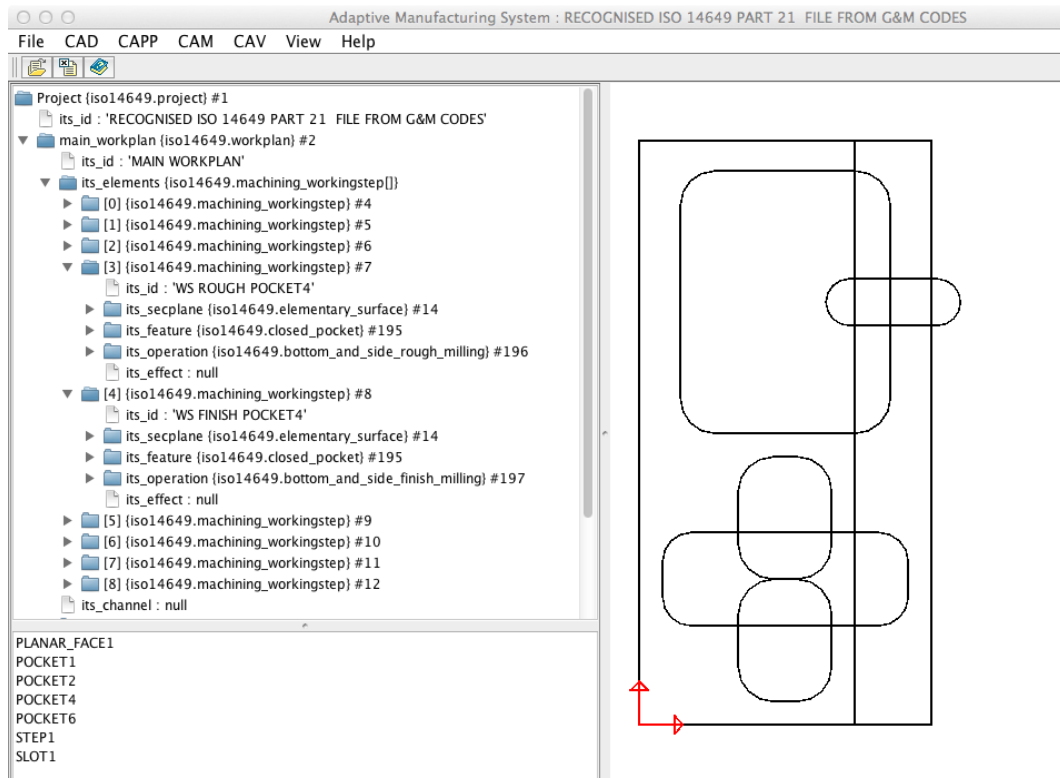


Figure 9.25 - Part III in the interoperable CAD/CAM system

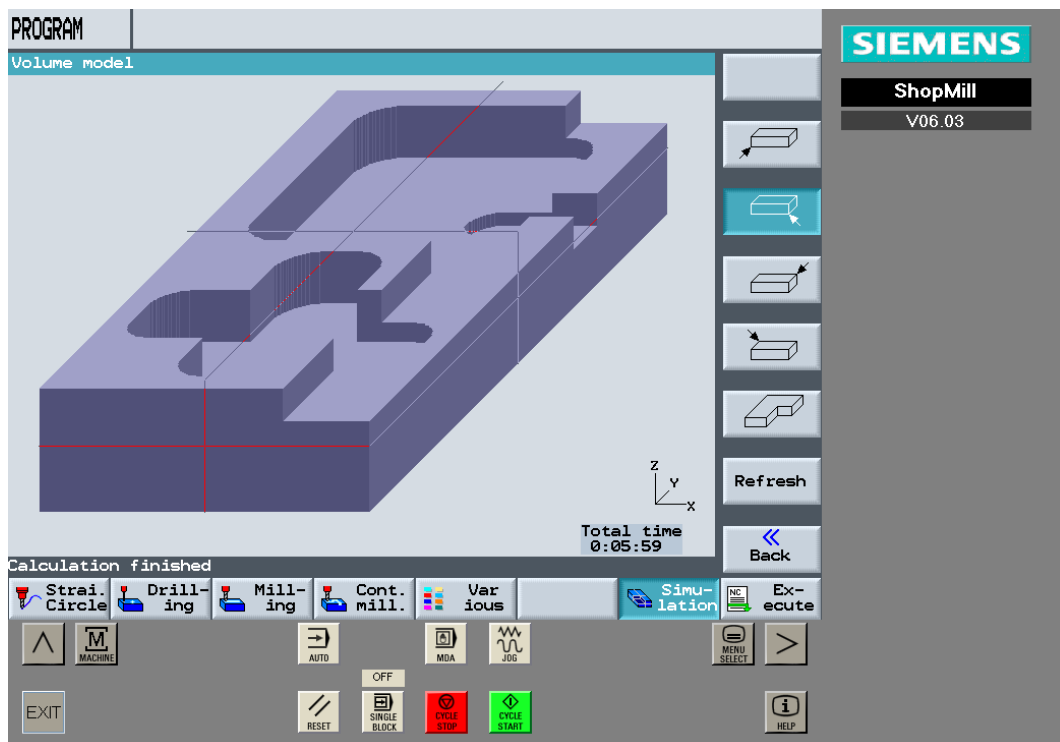


Figure 9.26 - Three dimensional simulation of Part III



Figure 9.27 - Part III machined on Siemens machine



Figure 9.28 - Test results for Part III

9.5. Results of the evaluations

From the results of the two tests in this chapter, the prototype of the universal process comprehension is capable of supporting different programming languages. The process comprehension method has been proven to be an effective approach to preserve the original process plan and reuse it on new manufacturing resources. The tests have demonstrated the application of universal process comprehension to realise the interoperability between one CAD/CAM system and two CNC machines. It suggests that utilising the prototype implementation, it is possible to achieve the scenario of the interoperability between a broader range of manufacturing systems to realise an interoperable manufacturing network.

10. Discussions, conclusions and future work

10.1. Introduction

This chapter provides a series of conclusions that have been derived as the result of the research. The contribution of this research has been highlighted together with the possible future area to explore.

10.2. Discussions

10.2.1. Interacting feature recognition from part programmes

In traditional feature recognition, the ability to handle interacting features has been an informal benchmark for industrial acceptability of a feature recognition systems (Vermaa and Rajotiab 2010). It is acknowledged as the most critical issue in the field.

One important reason for this situation is that, due to the feature interaction, only part of the feature surfaces or edges are left. It exposes huge challenges for feature recognition methods such as graph-based and hint-based feature recognition which rely on these traces. However, feature interaction in this research is not such a difficult issue any more, since the part programme is the process to create this feature(s) step by step. From the part programmes, the toolpath for each operation can be extracted. Hence, the material removed in each operation can be identified. It is quite similar to the volume decomposition methods. The difference is the volume machined off is the feature or the rough feature (rough operation). There is no need to combining volume cells to get meaningful features.

Although interacting features in this research is not a problem as the toolpath is used to recognise features, this method proposes other issues as well. The most critical issue is redundant features due to the fact that more than one operation can be applied on the same feature. The issue of feature identification has been discussed in Section 7.2.3.

10.2.2. Advantages of the universal process comprehension framework

Through universal process comprehension, a bidirectional information flow can be established from the shopfloor (CNC machines) with other manufacturing resources instead of a unidirectional information input to the shopfloor. It is quite essential since

for the first time the shopfloor is accommodated in the manufacturing chain for real: receiving information and feeding this back in an automatic way.

The information representation adopts the ISO 14649 STEP-NC standard which enables the possibility to share the information between various systems. It avoids the current issues of the incompatible data formats in the course of information exchange.

Connecting the shopfloor with the whole manufacturing chain enables knowledge recycle and reuse from the shopfloor. It fills the research gap of the shopfloor knowledge management and provides a reliable capture and reuse mechanism.

Another advantage of the realisation of the universal process comprehension framework is that there is no need to make modifications to the current manufacturing resources. The implementation of the framework into practice can be smoothly achieved.

10.2.3. Limitations of the universal process comprehension framework

There are a number of limitations associated with the universal process comprehension framework:

The ability to comprehend complex products geometry and the manufacturing knowledge such as machining strategies based on low-level information available in part programmes is still limited. Fully automatic comprehension still requires further development of the framework.

The proficiency of using a Meta-model to represent other none G&M codes part programmes is currently unproven. There are some other types of part programmes written in proprietary languages such as Heidenhain and Mazak. The ability utilising the XML schema proposed in this research to translating these languages into the meta-model has not been validated.

The framework proposed in the thesis is not capable in handling data relating to geometric tolerances and multiple setups or fixtures due to the limitations of information availability in NC part programmes. To support these functionalities, more comprehensive information and further improvement of the framework are needed.

Conclusions.

10.3. Conclusions

The conclusions formulated from this research are as follows:

- Shopfloor gap in terms of interoperability and knowledge reuse is not well addressed within the current literature.
- The architecture proposed in this research is viable for process comprehension to solve the interoperability issue and knowledge reuse problem at the shopfloor.
- The Meta-model of programming languages is capable of representing programming dialects to enable the process comprehension realised in a universal manner.
- The approach of feature recognition is adequate for 2½D part to support the architecture of universal process comprehension.
- STEP-NC provides a suitable data structure to accommodate the process plan and can be used as a neutral data format for interoperability between different systems. It is practical way to take advantage of the standard without modification to the existing resources.
- The prototype system developed in this research has shown the advantage of universal process comprehension for CNC interoperability and knowledge reuse utilising state-of-the-art commercial software and hardware tools.

10.4. Contribution to knowledge

The major contribution of this research to knowledge is the new vision of the shopfloor interoperability associated with process knowledge capture and reuse. It shows that process comprehension of part programmes can provide an effective solution to the issues of shopfloor interoperability and knowledge reuse in manufacturing industries. In addition, the proposed architecture and solution with UPCi and meta-model is viable for translating G&M codes across all CNC controller and machines. The approach of feature recognition is adequate for 2½D part to support the architecture of universal process comprehension.

10.5. Future work

Throughout the course of this research a number of opportunities to take this research further have been identified.

10.5.1. Integration of the resource modelling research

The universal process comprehension framework is mainly based on low-level part programmes, in which the process information is quite limited. Due to that reason, workpiece, cutting tool and programming specification information are necessary in this research. With integration of this research with the research on resource modelling (Vichare *et al.* 2009), the process comprehension framework can be realised in an automatic manner. The cutting tool and programming specification can be included in the resource model of CNC machines. More importantly, with comprehensive resources information available, the manufacturing knowledge can be accessed and adapted (such as changing the cutting tool) according to the target resource.

10.5.2. Extensibility to cover turning operations

The process comprehension framework proposed in this research has been proved on milling operations. However, it is applicable to turning operations as well, requiring research effort on new feature recognition algorithms for turning features.

10.5.3. Interfacing with other manufacturing systems

The framework presented can be integrated conveniently as an interface to other systems to be used in some other areas. Since the process information in the part programme is the latest and most accurate form of data for the final products, it is quite valuable to the systems or equipment which require real process information such as online inspection, shopfloor cost estimation system, reverse engineering system etc. Through process comprehension a shopfloor centred information network can be established to help manufacturing enterprises gain a competitive advantage.

10.5.4. Towards a universal manufacturing platform

The vision of a universal manufacturing platform by Newman and Nassehi (2007) envisaged to store standardised manufacturing information in a central manufacturing data warehouse, which is connected to various manufacturing systems, as shown in

Figure 10.1. Various systems can be linked together through the platform. UPCi can be implemented in the platform served as the CNC interface of the platform linking CNC machines. Through UPCi, the shopfloor knowledge can be shared across the design, manufacturing, market and service systems covering the whole product life cycle. The implementation UPCi will enable UMP more explicit.

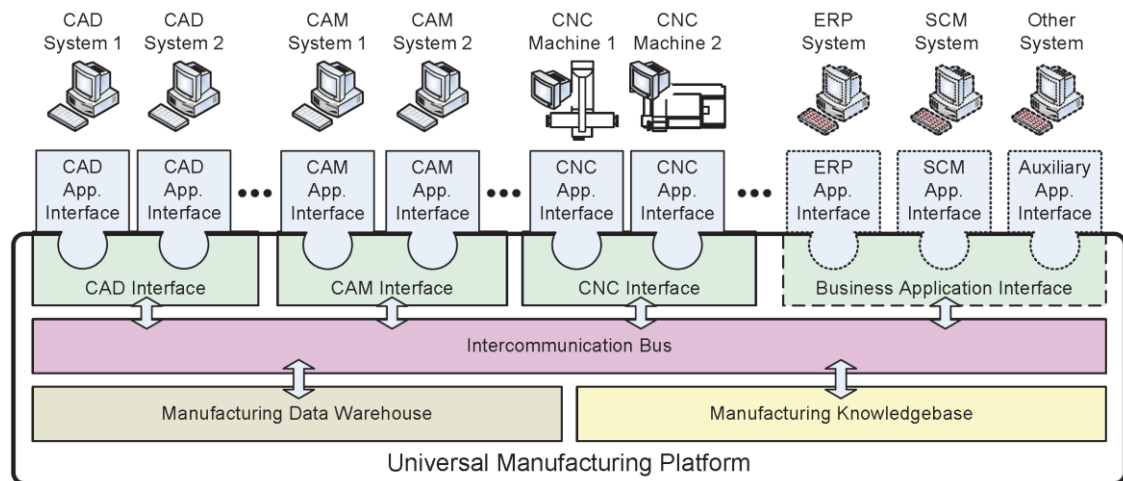


Figure 10.1 - Universal manufacturing platform (Newman and Nassehi 2007)

References

- Ali, L., Newman, S.T. and Petzing, J., 2005. Development of a STEP-compliant inspection framework for discrete components. *Proceedings of the Institution of Mechanical Engineers Part B-Journal of Engineering Manufacture*, 219(7), pp.557-563.
- Alizon, F., Shooter, S.B. and Simpson, T.W., 2006. Reuse of Manufacturing Knowledge to Facilitate Platform-Based Product Realization. *Journal of Computing and Information Science in Engineering*, 6(2), pp.170-178.
- Allen, R.D., 2003. An agent-based approach to STEP-NC CAD/CAM. Thesis (PhD). University of Loughborough, Loughborough, UK.
- Allen, R.D., Harding, J.A. and Newman, S.T., 2005. The application of STEP-NC using agent-based process planning. *International Journal of Production Research*, 43(4), pp.655 - 670.
- Alting, L., 1989. Computer aided process planning: the state-of-the-art survey. *International Journal of Production Research*, 27(4), pp.553.
- Amaitik, S. and Kiliç, S., 2007. An intelligent process planning system for prismatic parts using STEP features. *The International Journal of Advanced Manufacturing Technology*, 31(9), pp.978-993.
- Anjard, R.P., 1995. Computer integrated manufacturing: a dream becoming a reality. *Industrial Management & Data Systems*, 95(pp.3-4).
- Anwer, N. and Chep, A., 1999. IPPA: a knowledge assisted process planning system. *Proceedings of the 1999 7th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA '99)*, Barcelona, Spain. Vol.1, pp.297-301.
- Autodesk, 2012. Autodesk [Online]. Available from: Autodesk.com [Accessed on 2012].
- Balbontin, A., Yazdani, B.B., Cooper, R. and Souder, W.E., 2000. New product development practices in American and British firms. *Technovation*, 20(5), pp.257-274.
- Baxter, D., Gao, J., Case, K., Harding, J., Young, B., Cochrane, S. and Dani, S., 2007. An engineering design knowledge reuse methodology using process modelling. *Research in Engineering Design*, 18(1), pp.37-48.
- Baxter, D., Gao, J., Case, K., Harding, J., Young, B., Cochrane, S. and Dani, S., 2008. A framework to integrate design knowledge reuse and requirements management

in engineering design. *Robotics and Computer-Integrated Manufacturing*, 24(4), pp.585-593.

Bex, G.J., Neven, F. and Van den Bussche, J., 2004. DTDs versus XML schema: a practical study. *Proceedings of the Seventh International Workshop on theWeb and Databases (WebDB 2004)*, Paris, France. pp.79-84. 17-18 June.

Bhandarkar, M.P. and Nagi, R., 2000. STEP-based feature extraction from STEP geometry for Agile Manufacturing. *Computers in Industry*, 41(1), pp.3-24.

Borsellino, C., Licari, R., Lo Valvo, E. and Piacentini, M., 2004. An XML interface to STEP-NC systems for remote collaborative concurrent engineering. *Image Processing, Biomedicine, Multimedia, Financial Engineering and Manufacturing*, 18(pp.413-418.

Bourne, D., Corney, J. and Gupta, S.K., 2011. Recent Advances and Future Challenges in Automated Manufacturing Planning. *Journal of Computing and Information Science in Engineering*, 11(2), pp.021006.

Brandimarte, P. and Cantamessa, M., 1995. Methodologies for designing CIM systems: a critique. *Computers in Industry*, 25(3), pp.281-293.

Brecher, C., Vittr, M. and Wolf, J., 2006. Closed-loop CAPP/CAM/CNC process chain based on STEP and STEP-NC inspection tasks. *International Journal of Computer Integrated Manufacturing*, 19(6), pp.570 - 580.

Brunnermeier, S.B. and Martin, S.A., 1999. Interoperability cost analysis of the US automotive supply chain. Gaithersburg, MD: National Institute of Standards and Technology. (99-1).

Brunnermeier, S.B. and Martin, S.A., 2002. Interoperability costs in the US automotive supply chain. *Supply Chain management*, 7(2), pp.71-82.

Buzacott, J.A. and Yao, D.D., 1986. Flexible Manufacturing Systems: A Review of Analytical Models. *Management Science*, 32(7), pp.890-905.

Calabrese, F. and Celetanno, G., 2007. Design and realization of a STEP-NC compliant CNC embedded controller. *Proceedings of the 2007 IEEE Conference on Emerging Technologies and Factory Automation.*, Patras, Greece. 2007.

Campos, J.G. and Miguez, L.R., 2011. Standard process monitoring and traceability programming in collaborative CAD/CAM/CNC manufacturing scenarios. *Computers in Industry*, 62(3), pp.311-322.

Cang, G.L., Gui, G.S. and Wang, J.Q., 2006. STEP-NC- and XML-enabled e-manufacturing. *International Journal of Computer Applications in Technology*, 26(1/2), pp.59-64.

Cay, F. and Chassapis, C., 1997. An IT view on perspectives of computer aided process planning research. *Computers in Industry*, 34(3), pp.307-337.

CGTech, 2012. CGTech - the developer of VERICUT CNC simulation, optimization and verification software [Online]. Available from: <http://cgttech.com/> [Accessed on 9 January 2012].

Chen, H.H., Kang, H.-Y., Xing, X., Lee, A.H.I. and Tong, Y., 2008. Developing new products with knowledge management methods and process development management in a network. *Computers in Industry*, 59(2-3), pp.242-253.

Cheung, W.M. and Schaefer, D., 2009. *Product Lifecycle Management: State-of-the-art and Future Perspectives*, Enterprise Information Systems for Business Integration in SMEs: Technological, Organizational and Social Dimensions. Portugal: Polytechnic Institute of Cavado and Ave, pp.396.

Choi, B.K., 1984. Automatic recognition of machined surfaces from a 3D solid model. *Computer aided design*, 16(2), pp.81.

Chung, D.-H. and Suh, S.-H., 2008. ISO 14649-based nonlinear process planning implementation for complex machining. *CAD Computer Aided Design*, 40(5), pp.521-536.

Chung, J., Cook, R., Patel, D. and Simmons, M., 1988. Feature-based geometry construction for geometric reasoning. *Proceedings of the ASME-Computers Engineering Conference*, S. Fransisco.

Cochrane, S., Young, R., Case, K., Harding, J., Gao, J., Dani, S. and Baxter, D., 2008. Knowledge reuse in manufacturability analysis. *Robotics and Computer-Integrated Manufacturing*, 24(4), pp.508-513.

Curran, K. and Stenerson, J., 2001. *Computer numerical control : operation and programming*, 2nd ed. / Kelly Curran, Jon Stenerson. Upper Saddle River, N.J.: Prentice Hall.

Davenport, T.H. and Prusak, L., 1997. *Information Ecology:: Mastering the Information and Knowledge Environment*.

Delcam, 2012. Easy-to-use CAM software for milling machines and turn/mill centres [Online]. Available from: <http://www.featurecam.com/> [Accessed on 4 July 2012].

Denkena, B., Shpitalni, M., Kowalski, P., Molcho, G. and Zipori, Y., 2007. Knowledge management in process planning. *CIRP Annals-Manufacturing Technology*, 56(1), pp.175-180.

Dhupia, J., Powalka, B., Katz, R. and Ulsoy, A.G., 2007. Dynamics of the arch-type reconfigurable machine tool. *International Journal of Machine Tools and Manufacture*, 47(2), pp.326-334.

Ding, L., 2003. Feature technology and its applications in computer integrated manufacturing. Thesis (PhD). University of Luton, Luton.

Ding, L., Yue, Y., Ahmet, K., Jackson, M. and Parkin, R., 2005. Global optimization of a feature-based process sequence using GA and ANN techniques. *International Journal of Production Research*, 43(15), pp.3247-3272.

Donaldson, I.A. and Corney, J.R., 1993. Rule-based feature recognition for 2.5D machined components. *International Journal of Computer Integrated Manufacturing*, 6(1), pp.51-64.

ElMaraghy, H., 2005. Flexible and reconfigurable manufacturing systems paradigms. *International Journal of Flexible Manufacturing Systems*, 17(4), pp.261-276.

ElMaraghy, H., 2007. Reconfigurable Process Plans For Responsive Manufacturing Systems, *Digital Enterprise Technology*. Springer US, pp.35-44.

ElMaraghy, H.A., 1993. Evolution and Future Perspectives of CAPP. *CIRP Annals - Manufacturing Technology*, 42(2), pp.739-751.

ESPRIT, 2011. Universal Post Processing [Online]. Available from: <http://www.dpotechnology.com/en/products.asp> [Accessed on 2 Sep 2011].

Eversheim, W. and Schneewind, J., 1993. Computer-aided process planning--State of the art and future development. *Robotics and Computer-Integrated Manufacturing*, 10(1-2), pp.65-70.

Feeney, A.B., Kramer, T., Proctor, F., Hardwick, M. and Loffredo, D., 2003. STEP-NC Implementation - ARM or AIM? ISO T24 STEP-Manufacturing Meeting, S. Diego, USA. 2003-02-12.

Finin, T., Fritzson, R., McKay, D. and McEntire, R., 1994. KQML as an agent communication language. *Proceedings of the* pp.463.

Fischer, U. and Stokic, D., 2002. Organisational Knowledge Management in Manufacturing Enterprises-Solutions and Open Issues, Challenges and achievements in E-business and E-work. IOS Press, pp.819-826.

Fowler, J., 1995. STEP for data management, exchange and sharing, Twickenham: Technology Appraisals.

Gao, J.X., Aziz, H., Maropoulos, P.G. and Cheung, W.M., 2003. Application of product data management technologies for enterprise integration. *International Journal of Computer Integrated Manufacturing*, 16(7-8), pp.491.

Gao, S. and Shah, J.J., 1998. Automatic recognition of interacting machining features based on minimal condition subgraph. *Computer-Aided Design*, 30(9), pp.727-739.

Garetti, M., Terzi, S., Bertacci, N. and Brianza, M., 2005. Organisational change and knowledge management in PLM implementation. *International Journal of Product Lifecycle Management*, 1(1), pp.43-51.

GE Fanuc Automation, 1998. GE Fanuc Automation Operator's Manual Series 16i/18i/160i/180i - Model A, for Machining Center. GE Fanuc Automation.

GibbsCAM, 2012. Press Releases 2012: GibbsCAM Generates CNC Programs for Nakamura-Tome MTMs [Online]. Available from: http://www.gibbscam.com/news_events/index.php?page=press-releases&con=369 [Accessed on 17 June 2012].

Gouda, S. and Taraman, K., 1989. CAPP: AAST. present and future. *Proceedings of the SME International Conference and Exposition*, Detroit, MI. May 1-4.

Grabowski, R., 2002. CAD manager's guidebook, Albany, N.Y. ; Great Britain: OnWord Press.

Groover, M.P. and Zimmers, E.W., 1984. CAD/CAM: computer-aided design and manufacturing, London: Prentice-Hall.

Guo, X., Liu, Y., Du, D., Yamazaki, K. and Fujishima, M., 2011. A universal NC program processor design and prototype implementation for CNC systems. *The International Journal of Advanced Manufacturing Technology*, pp.1-15.

Han, J., Pratt, M. and Regli, W.C., 2000. Manufacturing feature recognition from solid models: a status report. *IEEE Transactions on Robotics and Automation*, 16(6), pp.782.

Hardwick, M. and Loffredo, D., 2006. Lessons learned implementing STEP-NC AP-238. *International Journal of Computer Integrated Manufacturing*, 19(6), pp.523-532.

Heidenhain, 2009. HEIDENHAIN TNCguide [Online]. Available from: <http://www.heidenhain.com/> [Accessed on 16 December 2009].

Henderson, M.R., 1984. Computer recognition and extraction of form features: a CAD/CAM link. *Computers in Industry*, 5(4), pp.329-339.

Hicks, B.J., Culley, S.J., Allen, R.D. and Mullineux, G., 2002. A framework for the requirements of capturing, storing and reusing information and knowledge in engineering design. *International Journal of Information Management*, 22(4), pp.263-280.

IEEE, 1990. IEEE standard computer dictionary. A compilation of IEEE standard computer glossaries, IEEE Std 610. N.Y.: The Institution of Electrical and Electronics Engineers.

ISO6983-1, 1982. Numerical control of machines—program format and definition of address words, Part 1: data format for positioning, Line motion and contouring control systems. International Standards Organisation (ISO).

ISO 10303-11, 1994. Industrial automation systems and integration - product data representation and exchange, part 11: Description methods: Express language reference manual. International Standards Organisation (ISO).

ISO 10303-21, 2002. Industrial automation systems and integration - product data representation and exchange, Part 21: Implementation methods: Clear text encoding of the exchange structure. International Standards Organisation (ISO).

ISO 10303-28, 2007. Industrial automation systems and integration - product data representation and exchange, Part 28: Implementation methods: XML representations of EXPRESS schemas and data, using XML schemas. International Standards Organisation (ISO).

ISO 10303-42, 2003. Industrial automation systems and integration - product data representation and exchange, part 42: Integrated Generic Resource: Geometric and Topological Representation. International Standards Organisation (ISO).

ISO 10303-224, 2000. Industrial automation systems and integration - product data representation and exchange, part 224: Application protocol: Mechanical product definition for process plans using machining features. International Standards Organisation (ISO).

ISO 10303-514, 1994. Industrial automation systems and integration - product data representation and exchange, Part 514: Application interpreted construct: Advanced boundary representation. International Standards Organisation (ISO).

ISO 14649-1, 2002. Industrial automation systems and integration - physical device control - data model for computerized numerical controllers, part 1: Overview and fundamental principles. International Standards Organisation (ISO).

ISO 14649-10, 2002. Industrial automation systems and integration - physical device control - data model for computerized numerical controllers, part 1: General process data. International Standards Organisation (ISO).

ISO 14649-11, 2002. Industrial automation systems and integration - physical device control - data model for computerized numerical controllers, part 11: Process data for milling. International Standards Organisation (ISO).

ISO 14649-12, 2004. Industrial automation systems and integration - physical device control - data model for computerized numerical controllers, part 12: Process data for turning. International Standards Organisation (ISO).

ISO 14649-111, 2002. Industrial automation systems and integration - physical device control - data model for computerized numerical controllers, part 11: Tools for milling. International Standards Organisation (ISO).

ISO 14649-121, 2002. Industrial automation systems and integration - physical device control - data model for computerized numerical controllers, part 121: Tools for turning machines. International Standards Organisation (ISO).

Jardim-Goncalves, R., Figay, N. and Steiger-Garcia, A., 2006. Enabling interoperability of STEP Application Protocols at meta-data and knowledge level. *International Journal of Technology Management*, 36(4), pp.402-421.

Joshi, S., 1988. Graph-based heuristics for recognition of machined features from a 3 D solid model. *Computer aided design*, 20(2), pp.58-66.

Kailash and B., 2000. Multiple feature interpretation across domains. *Computers in Industry*, 42(1), pp.13-32.

Kern, V. and Bohn, J., 1997. STEP databases for product data exchange. *Proceedings of the Vol.3*, pp.1337-1341.

Kim, Y., Kang, S.-H., Lee, S.-H. and Yoo, S.B., 2001. A distributed, open, intelligent product data management system. *International Journal of Computer Integrated Manufacturing*, 14(2), pp.224-235.

Kramer, T.R., Proctor, F., Xu, X. and Michaloski, J.L., 2006. Run-time interpretation of STEP-NC: implementation and performance. *International Journal of Computer Integrated Manufacturing*, 19(6), pp.495-507.

Kyprianou, L.K., 1980. Shape classification in computer aided design. Thesis (Ph.D). Univ. Cambridge, Cambridge, UK.

Laakko, T., 1993. Feature modelling by incremental feature recognition. *Computer aided design*, 25(8), pp.479-492.

Lam, S.M. and Wong, T.N., 2000. Recognition of machining features - a hybrid approach International Journal of Production Research, 38(17), pp.4301-4316.

Lee, C.Y., 1993. A Recent Development of the Integrated Manufacturing System: A Hybrid of MRP and JIT. International Journal of Operations & Production Management, 13(4), pp.3-17.

Lee, D. and Chu, W.W., 2012. Comparative Analysis of Six XML Schema Languages [Online]. Available from: <http://www.cobase.cs.ucla.edu/tech-docs/dongwon/sigmod-record-00.html> [Accessed on 20 June 2012].

Lee, W. and Bang, Y.-B., 2003. Design and implementation of an ISO14649-compliant CNC milling machine. International Journal of Production Research, 41(13), pp.3007-3017.

Leitão, P. and Restivo, F., 2006. ADACOR: A holonic architecture for agile and adaptive manufacturing control. Computers in Industry, 57(2), pp.121-130.

Li, P., Hu, T. and Zhang, C., 2012. Development of an ontology-based and STEP-compliant intelligent CNC system. Applied Mechanics and Materials. Proceedings of the 2nd International Conference on Functional Manufacturing and Mechanical Dynamics, January 22, 2012 - January 25, 2012, Hangzhou, Zhejiang, China. Vol.141, pp.251-257.

Lin, S.-C.J., 1994. Computer numerical control : essentials in programming and networking, Albany, N.Y.: Delmar Publishers.

Lindholm, T. and Yellin, F., 1999. The Java™ Virtual Machine Specification, Second Edition. Available from: [http://java.sun.com/docs/books/jvms/second edition/html/VMSpecTOC.doc.html](http://java.sun.com/docs/books/jvms/second%20edition/html/VMSpecTOC.doc.html) [Accessed on 17 October 2011].

Liu, T.D. and Xu, W.X., 2001. A review of web-based product data management systems. Computers in Industry, 44(3), pp.251-262.

Liu, Y., Guo, X., Li, W., Yamazaki, K., Kashiara, K. and Fujishima, M., 2007. An intelligent NC program processor for CNC system of machine tool. Robotics and Computer-Integrated Manufacturing, 23(2), pp.160-169.

Loffredo, D., 2000. Fundamentals of STEP Implementation [Online]. Available from: <http://www.steptools.com/library/fundimpl.pdf> [Accessed on 17 October 2011].

Maeder, W., Nguyen, V.K., Richard, J. and Stark, J., 2002. Standardisation of the Manufacturing Process: the IMS STEP-NC project. Proceedings of the INational Network of Competence on Integrated Production Logistics Workshop, Saas Fee, Switzerland. pp.5.5/1-5.5/3. Sep. 10-12.

Manufacture, 2004. Manufacture: A vision for 2020, Assuring the future of manufacturing in Europe. Report of the High-Level Group.

Márkus, A., Váncza, J. and Horváth, M., 1997. Process planning by retrieval and adaptation. *Computers in Industry*, 33(1), pp.47-60.

Maropoulos, P., 1995a. A novel process planning architecture for product-based manufacture. *Proceedings of the Institution of Mechanical Engineers. Pt. B. Journal of Engineering Manufacture*, 209(pp.267-76.

Maropoulos, P.G., 1995b. Review of research in tooling technology, process modelling and process planning part II: Process planning. *Computer Integrated Manufacturing Systems*, 8(1), pp.13-20.

Martin, R.A., 2005. International standard for system integration. International Concil on System Engineering, Rochester, NY.

Mazak, Y., 2010. Production Support Software, CAD/CAM: CAMWARE [Online]. Available from: <http://english.mazak.jp/products/system/camware.html> [Accessed on 17 October 2011].

McMahon, C. and Browne, J., 1993. CAD/CAM from Principle to Practice, Addison Wesley.

Mehrabi, M.G., Ulsoy, A.G. and Koren, Y., 2000. Reconfigurable manufacturing systems: Key to future manufacturing. *Journal of Intelligent Manufacturing*, 11(4), pp.403-419.

Miao, H.K., Sridharan, N. and Shah, J.J., 2002. CAD-CAM integration using machining features. *International Journal of Computer Integrated Manufacturing*, 15(4), pp.296-318.

Ming, X. and Mak, K., 2000. A hybrid Hopfield network-genetic algorithm approach to optimal process plan selection. *International Journal of Production Research*, 38(8), pp.1823-1839.

MIS Group, 2011. Predator OutPost - Universal Post & Reverse Post Translator [Online]. Available from: <http://www.mis-group.com/outpost/outpost.php> [Accessed on Sep 2 2011].

Mittelmann, A., 2005. Knowledge Sharing on the Shop Floor - a Critical Success Factor for Vital Companies. voestalpine Stahl GmbH: Division Stal.

Mokhtar, A. and Houshmand, M., 2010. Introducing a roadmap to implement the Universal Manufacturing Platform using Axiomatic Design theory. *International Journal of Manufacturing Research*, 5(2), pp.252-269.

Mun, D. and Ramani, K., 2011. Knowledge-based part similarity measurement utilizing ontology and multi-criteria decision making technique. *Advanced Engineering Informatics*, 25(2), pp.119-130.

Nassehi, A., 2007. The realisation of CAD/CAM/CNC interoperability in prismatic part manufacturing. University of Bath, Bath, UK.

Nassehi, A., Allen, R.D. and Newman, S.T., 2006a. Application of mobile agents in interoperable STEP-NC compliant manufacturing. *International Journal of Production Research*, 44(18), pp.4159 - 4174.

Nassehi, A., Liu, R. and Newman, S.T., 2007. A new software platform to support feature-based process planning for interoperable STEP-NC manufacture. *International Journal of Computer Integrated Manufacturing*, 20(7), pp.669-683.

Nassehi, A., Newman, S.T. and Allen, R.D., 2006b. The application of multi-agent systems for STEP-NC computer aided process planning of prismatic components. *International Journal of Machine Tools and Manufacture*, 46(5), pp.559-574.

National Coalition for Advanced manufacturing, 2001. Exploiting E-Manufacturing: Interoperability of Softwares Used by U.S. manufacturers, Washington D.C.:

NCCS, 2011. PostWorks Universal Postprocessor [Online]. Available from: http://www.nccs.com/pages/Products_main.html [Accessed on Sep 2 2011].

Newman, S.T., 1982. Design of curved surface. Thesis (BSc). University of Aston, Birmingham, UK.

Newman, S.T., Allen, R.D. and Rosso Jr, R.S.U., 2003. CAD/CAM solutions for STEP-compliant CNC manufacture. *International Journal of Computer Integrated Manufacturing*, 16(7-8), pp.590-597.

Newman, S.T. and Nassehi, A., 2007. Universal Manufacturing Platform for CNC Machining. *CIRP Annals - Manufacturing Technology*, 56(1), pp.459-462.

Newman, S.T., Nassehi, A., Xu, X.W., Rosso Jr, R.S.U., Wang, L., Yusof, Y., Ali, L., Liu, R., Zheng, L.Y., Kumar, S., Vichare, P. and Dhokia, V., 2008. Strategic advantages of interoperability for global manufacturing using CNC technology. *Robotics and Computer-Integrated Manufacturing*, 24(6), pp.699-708.

Nezis, K., 1997. Recognizing 2 1/2 D shape features using a neural network and heuristics. *Computer aided design*, 29(7), pp.523.

Nguyen, V. and Stark, J., 2005. STEP-compliant CNC Systems, Present and Future Directions, *Advanced Design and Manufacturing Based on STEP*. Springer, pp.215-231.

Niebel, B., 1965. Mechanized process selection for planning new designs. ASME paper, No. 737.

Nonaka, I. and Takeuchi, H., 1995. The knowledge-creating company: How Japanese companies create the dynamics of innovation, Oxford University Press, USA.

Ong, S.K., 2002. An internet-based virtual CNC milling system. International Journal of Advanced Manufacturing Technology, 20(1), pp.20-30.

Oracle Corporation, 2011. NetBeans [Online]. Available from: <http://netbeans.org/> [Accessed on 21 October 2011].

Peng, Y., Finin, T., Labrou, Y., Cost, R.S., Chu, B., Long, J., Tolone, W.J. and Boughannam, A., 1999. Agent-based approach for manufacturing integration: the CIIMPLEX experience. Applied artificial intelligence, 13(1/2), pp.39-63.

Platts, K., 1995. Integrated manufacturing: a strategic approach. Integrated Manufacturing Systems, 6(3), pp.18-23.

Prabhakar, S., 1992. Automatic form-feature recognition using neural-network-based techniques on boundary representations of solid models. Computer aided design, 24(7), pp.381-393.

Predator Software, 2012. Predator Software - DNC, MDC, oEE, PDM, SFC, Virtual CNC, RCM, Traveler, Tracker Tool or Gage Crib manufacturing systems [Online]. Available from: <http://www.predator-software.com> [Accessed on 9 January 2012].

Proctor, F.M., Kramer, T.R. and Michaloski, J.L., 1997. Canonical machining commands, NISTIR 5970. ISD of NIST, USA.

Proctor, F.M., Michaloski, J.L. and Shackleford, W.P., 2002. Tying together design, process planning and machining with STEP-NC technology. Proceedings of the 5th Biannual World Automation Congress, Orlando, Fl. pp.33-38. Jun 09-13.

Pusztai, J. and Sava, M., 1983. Computer numerical control, Reston, Va.: Reston Publishing.

Rauch, M., Laguionie, R., Hascoet, J.-Y. and Suh, S.-H., 2012. An advanced STEP-NC controller for intelligent machining processes. Robotics and Computer-Integrated Manufacturing, 28(3), pp.375-384.

Rauch, M., Laguionie, R., Hascoet, J.Y. and Xu, X., 2009. Enhancing CNC Manufacturing Interoperability with STEP-NC. Journal of Machine Engineering, 9(4), pp.26-37.

Ray, S. and Jones, A., 2006. Manufacturing interoperability. *Journal of Intelligent Manufacturing*, 17(6), pp.681-688.

Reed, N., Scanlan, J., Wills, G. and Halliday, S.T., 2011. Knowledge use in an advanced manufacturing environment. *Design Studies*, 32(3), pp.292-312.

Rosso Jr, R.S.U., 2005. STEP compliant CAD/CAPP/CAM system for rotational asymmetric parts. Loughborough University, Loughborough, UK.

Schroeder, C., 1998. Printed circuit board design using autoCAD, EDN series for design engineers. Newnes.

Schroeder, T. and Hoffmann, M., 2006. Flexible automatic converting of NC programs. A cross-compiler for structured text. *International Journal of Production Research*, 44(13), pp.2671-2679.

Shah and Mantyla, 1995. Parametric and feature-based CAD/CAM: concepts, techniques, and applications, John Wiley & Sons.

Shen, W., 2005. iShopFloor: an Internet-enabled agent-based intelligent shop floor. *IEEE transactions on systems, man and cybernetics. Part C, Applications and reviews*, 35(3), pp.371-381.

Shin, S.-J., Um, J., Yoon, J.-S., Jeong, S., Cha, J.-M., Suh, S.-H. and Chung, D.-H., 2009. Developing ISO 14649-based conversational programming system for multi-channel complex machine tools. *International Journal of Computer Integrated Manufacturing*, 22(6), pp.562 - 575.

Shin, S.J., Suh, S.H. and Stroud, I., 2007. Reincarnation of G-code based part programs into STEP-NC for turning applications. *Computer-Aided Design*, 39(1), pp.1-16.

Siemens, 2009. ShopMill-Motion Control System [Online]. Available from: <http://www.automation.siemens.com/> [Accessed on 16 December 2009].

Siemens AG, 2000. Short Guide Programming: SINUMERIK 840D/840Di 810D/FM-NC, Siemens AG.

Smid, P., 2003. CNC programming handbook : a comprehensive guide to practical CNC programming, 2nd ed. New York: Industrial Press.

Spence, A.D., Abrari, F. and Elbestawi, M.A., 2000. Integrated solid modeller based solutions for machining. *Computer-Aided Design*, 32(8-9), pp.553-568.

STEP Tools Inc, 2004. STEP Index Library (STIX) [Online]. Available from: <http://www.steptools.com/stix/> [Accessed on 6 March 2009].

STEP Tools Inc, 2012. STEP AP 238 FAQ: Questions about Manufacturing Data Standards [Online]. Available from: http://www.steptools.com/library/stepnc/faq/faq_02.html [Accessed on 25 June 2012].

Stute, G. and Nann, R., 1971. Report on a Special DNC System. Proceedings of the 12th International Conference on Machine Tool Design and Research, pp.429-432.

Suh, S.H. and Cheon, S.U., 2002. A framework for an intelligent CNC and data model. International Journal of Advanced Manufacturing Technology, 19(10), pp.727-735.

Suh, S.H., Cho, J.H. and Hong, H.D., 2002a. On the architecture of intelligent STEP-compliant CNC. International Journal of Computer Integrated Manufacturing, 15(2), pp.168 - 177.

Suh, S.H., Chung, D.H., Lee, B.E., Cho, J.H., Cheon, S.U., Hong, H.D. and Lee, H.S., 2002b. Developing an integrated STEP-compliant CNC prototype. Journal of Manufacturing Systems, 21(5), pp.350-362.

Suh, S.H. and Lee, B.E., 2004. STEP-manufacturing roadmap. Proceedings of the Proceedings of Korea CAD/CAM conference, Kangwon, Korea.

Suh, S.H., Lee, B.E., Chung, D.H. and Cheon, S.U., 2003. Architecture and implementation of a shop-floor programming system for STEP-compliant CNC. Computer-Aided Design, 35(12), pp.1069-1083.

Tissen, R., Andriessen, D. and Deprez, F.L., 1998. Creating the 21st Century Company: Knowledge Intensive, People Rich, Value-Based Knowledge Management, Addison Wesley Longman.

Toh, K.T.K. and Newman, S.T., 1996. The future role of DNC in metalworking SMEs. International Journal of Production Research, 34(3), pp.863-877.

Tsuji, M., 2003. Technological innovation and the formation of Japanese technology: the case of the machine tool industry. AI & Society, 17(3), pp.291-306.

Tuomi, I., 1999. Data is More Than Knowledge: implications of the reversed knowledge hierarchy for knowledge management and organizational memory. Journal of Management Information Systems, 16(3), pp.107-121.

Vandenbrande, J.H. and Requicha, A.A.G., 1993. Spatial Reasoning for the Automatic Recognition of Machinable Features in Solid Models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(12), pp.1269-1285.

- Vermaa, A. and Rajotiab, S., 2010. A review of machining feature recognition methodologies. *International Journal of Computer Integrated Manufacturing*, 23(4), pp.353-368.
- Vichare, P., Nassehi, A., Kumar, S. and Newman, S.T., 2009. A Unified Manufacturing Resource Model for representing CNC machining systems. *Robotics and Computer-Integrated Manufacturing*, 25(6), pp.999-1007.
- W3C,2012. XML Schema [Online]. Available from: <http://www.w3.org/XML/Schema> [Accessed on 14 March 2012].
- W3C Recommendation,2008. Extensible Markup Language (XML) 1.0 (Fifth Edition) [Online]. Available from: <http://www.w3.org/TR/2008/REC-xml-20081126/> [Accessed on 4 December 2009].
- Wang, E. and Kim, Y.S., 1998. Form feature recognition using convex decomposition: results presented at the 1997 ASME CIE Feature Panel Session. *Computer-Aided Design*, 30(13), pp.983-989.
- Wang, J.X., Tang, M.X., Song, L.N. and Jiang, S.Q., 2009. Design and implementation of an agent-based collaborative product design system. *Computers in Industry*, 60(7), pp.520-535.
- Wang, L., Cai, N., Feng, H.Y. and Liu, Z., 2006. Enriched machining feature-based reasoning for generic machining process sequencing. *International Journal of Production Research*, 44(8), pp.1479-1501.
- Wang, L. and Shen, W., 2003. DPP: an agent-based approach for distributed process planning. *Journal of Intelligent Manufacturing*, 14(5), pp.429-439.
- Wesley, M.A. and Markowsky, G., 1981. Fleshing Out Projections. *IBM Journal of Research and Development*, 25(6), pp.934-954.
- Wiig, K.M., 1993. *Knowledge management foundations: thinking about thinking: how people and organizations create, represent, and use knowledge*, Schema Press.
- Wolf, J., 2003. Requirements in NC machining and use cases for STEP-NC—analysis of ISO 14 649 (ARM) and AP 238 (AIM). ISO T24 STEP-Manufacturing Meeting, S. Diego, USA.
- Woo, Y. and Sakurai, H., 2002. Recognition of maximal features by volume decomposition. *Computer-Aided Design*, 34(3), pp.195-207.
- Wosnik, M., Kramer, C., Selig, A. and Klemm, P., 2006. Enabling feedback of process data by use of STEP-NC. *International Journal of Computer Integrated Manufacturing*, 19(6), pp.559 - 569.

- Xie, S.Q. and Xu, X., 2006. A STEP-compliant process planning system for sheet metal parts. *International Journal of Computer Integrated Manufacturing*, 19(6), pp.627 - 638.
- Xu, X. and He, Q., 2002. STEP-NC to Re-shape Manufacturing Industry. *Proceedings of the Proceedings of the 5th International Conference on Frontiers of Design and Manufacturing (ICFDM'2002)*, Dalian, China. pp.125-131.
- Xu, X., Wang, L. and Newman, S.T., 2011. Computer-aided process planning - A critical review of recent developments and future trends. *International Journal of Computer Integrated Manufacturing*, 24(1), pp.1-31.
- Xu, X.W., 2006a. Realization of STEP-NC enabled machining. *Robotics and Computer-Integrated Manufacturing*, 22(2), pp.144-153.
- Xu, X.W., 2006b. STEP-NC and function blocks for interoperable manufacturing. *IEEE Transactions on Automation Science and Engineering*, 3(3), pp.297-308.
- Xu, X.W. and He, Q., 2004. Striving for a total integration of CAD, CAPP, CAM and CNC. *Robotics and Computer-Integrated Manufacturing*, 20(2), pp.101-109.
- Xu, X.W. and Newman, S.T., 2006. Making CNC machine tools more open, interoperable and intelligent - A review of the technologies. *Computers in Industry*, 57(2), pp.141-152.
- Xu, X.W., Wang, H., Mao, J., Newman, S.T., Kramer, T.R., Proctor, F.M. and Michaloski, J.L., 2005. STEP-compliant NC research: The search for intelligent CAD/CAPP/CAM/CNC integration. *International Journal of Production Research*, 43(17), pp.3703-3743.
- Zhang, C., Liu, R. and Hu, T., 2006. On the futuristic machine control in a STEP-compliant manufacturing scenario. *International Journal of Computer Integrated Manufacturing*, 19(6), pp.508-515.
- Zhang, c., Liu, r. and Wang, h., 2002. Latest Development Trend of CNC Technology-STEP-NC. *Manufacturing Technology & Machine Tool*.
- Zhang, R. and Zhou, X., 2011. Similarity Assessment of Mechanical Parts Based on Integrated Product Information Model. *Journal of Computing and Information Science in Engineering*, 11(1), pp.011001(1-12).
- Zhang, X., Liu, R., Nassehi, A. and Newman, S.T., 2011a. A STEP-compliant process planning system for CNC turning operations. *Robotics and Computer-Integrated Manufacturing*, 27(2), pp.349-356.

Zhang, Y., Huang, G.Q., Qu, T. and Ho, O., 2010. Agent-based workflow management for RFID-enabled real-time reconfigurable manufacturing. *International Journal of Computer Integrated Manufacturing*, 23(2), pp.101-112.

Zhang, Y., Huang, G.Q., Qu, T., Ho, O. and Sun, S., 2011b. Agent-based smart objects management system for real-time ubiquitous manufacturing. *Robotics and Computer-Integrated Manufacturing*, 27(3), pp.538-549.

Zhao, Y., Habeeb, S. and Xu, X., 2009. Research into integrated design and manufacturing based on STEP. *The International Journal of Advanced Manufacturing Technology*, 44(5), pp.606-624.

Zheng, L., Dong, H., Vichare, P., Nassehi, A. and Newman, S.T., 2008. Systematic modeling and reusing of process knowledge for rapid process configuration. *Robotics and Computer-Integrated Manufacturing*, 24(6), pp.763-772.

Zhou, Y.J., 2004. An empirical study of shop floor tacit knowledge acquisition in Chinese manufacturing enterprises. *International Journal of Industrial Ergonomics*, 34(4), pp.249-261.

Appendix A. Publications

1. **Zhang, X.**, Nassehi, A. and Newman, S.T.

Process comprehension for interoperable CNC manufacturing,
Proceedings of the Computer Science and Automation Engineering (CSAE),
2011 IEEE International Conference on, Vol.4, pp.225-229. 10-12 June 2011.

Over the last 40 years manufacturing industry has enjoyed a rapid growth with the support of CNC machines and various computer-aided systems (CAD, CAPP, CAM etc.) known collectively as CAx systems. Interoperability across different CAx systems has become increasingly important for manufacturing industries, especially in the current competitive marketing environment. Even though CNC machines are major contributors to the production capacity of enterprises, the interoperability between CNC machines and both CAx systems and other CNC machines remains difficult due to the low level part programming language that is used (G&M codes) to programme the machines. In this research, a Universal Process Comprehension interface (UPCi) for different CNC machines is proposed aiming to comprehend a system generic process plan from a low-level part programme. This would enable shopfloor interoperability and allows bidirectional communication between CNC machines and CAx resources. The novel architecture proposed in this research is extensible to accommodate different types of CNCs. A new type of CNC can be interfaced with the system by defining a semantic dictionary of its programming rules. In this paper, the framework of UPCI is introduced. A prototype implementation is presented and the method is verified through the use of an example case study part.

2. **Zhang, X.**, Nassehi, A., Dhokia, V.G. and Newman, S.T.

Refining Process Logic From CNC Part Programmes for Integrated STEP-NC Compliant Manufacturing of Prismatic Parts,
Proceedings of the 4th International Conference on Changeable, Agile, Reconfigurable and Virtual Production (CARV 2011), Montreal, Canada. pp.333-338. 2-5 October 2011.

Reusing machining process information offers the industry opportunities to improve manufacturing traceability, standardization, quality, and control while enabling savings in cost and time. Rapid product development relies heavily on quick and reliable process planning and knowledge reuse to facilitate the generation of process plans efficiently and effectively. STEP-NC as a STEP compliant standard for CNC manufacturing provides a promising opportunity for knowledge reuse and information sharing across the CAx manufacturing chain from CAD to CNC. However, the widespread use of G&M codes impedes the feedback of shopfloor knowledge. In this research, to capture shopfloor knowhow, a Universal Process Comprehension interface (UPCi) is utilised to comprehend the process plan from low-level G&M code programmes written for CNC machines and represent it in a standardised STEP-NC format. In this paper, a novel method is proposed to capture the machining process knowledge from CNC part programmes with UPCI and reuse it on new manufacturing resources. An example part is used as a case study to illustrate the process plan comprehension and how the same process plan can be utilised to manufacture the product using new resources.

Appendix B. XML specifications of Fanuc and Siemens programming dialects

B.1. XML specifications of Fanuc 18i

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  Document : Siemens.xml
  Created on : 02 November 2009, 11:00
  Author : Xianzhi Zhang
  Description:
    G&M codes programming specification of Fanuc controllers.
-->

<root xmlns="http://www.w3schools.com">
  <Make>Fanuc18i</Make>
  <Version>X3</Version>
  <Author>Xianzhi Zhang</Author>
  <Date>2011-10-31</Date>

  <Grammar>
    <Comments>
      <Start>{</Start>
      <End>}</End>
    </Comments>
  </Grammar>
  <Position>
    <Parameters>
      <Mandatory>
        <Prm>$X@axis0</Prm>
        <Prm>$Y@axis1</Prm>
        <Prm>$Z@axis2</Prm>
      </Mandatory>
    </Parameters>
  </Position>
  <Movements>
    <RapidPosition>
      <Modal>true</Modal>
      <Syntax>
        <Format>$*G0*$X@axis0$Y@axis1$Z@axis2</Format>
        <Parameters>
          <Optional>
            <Prm>$G00</Prm>
          </Optional>
          <Mandatory/>
        </Parameters>
      </Syntax>
    </RapidPosition>
```

```

<LinearInterpolation>
  <Modal>true</Modal>
  <Syntax>
    <Format>*$G1*$X@axis0$Y@axis1$Z@axis2$F@feedrate</Format>
    <Parameters>
      <Optional>
        <Prm>$G1</Prm>
        <Prm>$F@feedrate</Prm>
      </Optional>
      <Mandatory/>
    </Parameters>
  </Syntax>
</LinearInterpolation>
<CWCIRCULARInterpolation>
  <Modal>true</Modal>
  <Syntax>

<Format>*$G2*$X@axis0$Y@axis1$Z@axis2$R@radius$H@helical$F@feedrate</Form
at>
    <Parameters>
      <Optional>
        <Prm>$G2</Prm>
        <Prm>$R@radius</Prm>
        <Prm>$H@helical</Prm>
        <Prm>$F@feedrate</Prm>
      </Optional>
      <Mandatory>
        <Prm>$R@radius</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
  <Syntax>

<Format>*$G2*$X@axis0$Y@axis1$Z@axis2$I@R1$J@R2$K@R3$H@helical$F@feedra
te</Format>
    <Parameters>
      <Optional>
        <Prm>$G@02</Prm>
        <Prm>$I@cx</Prm>
        <Prm>$J@cy</Prm>
        <Prm>$K@cz</Prm>
        <Prm>$H@helical</Prm>
        <Prm>$F@feedrate</Prm>
      </Optional>
      <Mandatory/>
    </Parameters>
  </Syntax>
</CWCIRCULARInterpolation>

<CCWCIRCULARInterpolation>
  <Modal>true</Modal>
  <Syntax>

```

```
<Format>*$G3*$X@axis0$Y@axis1$Z@axis2$R@radius$H@helical$F@feedrate</Format>
```

```

    <Parameters>
      <Optional>
        <Prm>$G03</Prm>
        <Prm>$R@radius</Prm>
        <Prm>$H@helical</Prm>
        <Prm>$F@feedrate</Prm>
      </Optional>
      <Mandatory>
        <Prm>$R@radius</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
<Syntax>
```

```
<Format>*$G3*$X@axis0$Y@axis1$Z@axis2$I@R1$J@R2$K@R3$H@helical$F@feedrate</Format>
```

```

    <Parameters>
      <Optional>
        <Prm>$G@03</Prm>
        <Prm>$I@cx</Prm>
        <Prm>$J@cy</Prm>
        <Prm>$K@cz</Prm>
        <Prm>$H@helical</Prm>
        <Prm>$F@feedrate</Prm>
      </Optional>
      <Mandatory/>
    </Parameters>
  </Syntax>
```

```
</CCWCIRCULARInterpolation>
```

```
<CycleCancel>
```

```
  <Modal>true</Modal>
```

```
  <Syntax>
```

```
    <Format>*$G80*</Format>
```

```
    <Parameters>
```

```
      <Mandatory>
```

```
        <Prm>$G80</Prm>
```

```
      </Mandatory>
```

```
    </Parameters>
```

```
  </Syntax>
```

```
</CycleCancel>
```

```
<SpotDrilling>
```

```
  <Modal>true</Modal>
```

```
  <Syntax>
```

```
<Format>*$G81*$X@axis0$Y@axis1$Z@axis2$R@hight$F@feedrate$K@repeat</Format>
```

```
  <Parameters>
```

```
    <Optional>
```

```

        <Prm>$G81</Prm>
        <Prm>$F@feedrate</Prm>
        <Prm>$K@repeat</Prm>
    </Optional>
    <Mandatory>
        <Prm>$R@StartPoint</Prm>
        <Prm>$Z@Depth</Prm>
    </Mandatory>
</Parameters>
</Syntax>
</SpotDrilling>
<CounterBoring>
    <Modal>true</Modal>
    <Syntax>

```

```

<Format>$*G82*$X@axis0$Y@axis1$Z@axis2$R@hight$P@dwell$F@feedrate$K@repe
at</Format>

```

```

    <Parameters>
        <Optional>
            <Prm>$G82</Prm>
            <Prm>$F@feedrate</Prm>
            <Prm>$K@repeat</Prm>
        </Optional>
        <Mandatory>
            <Prm>$P@dwell</Prm>
            <Prm>$R@StartPoint</Prm>
            <Prm>$Z@Depth</Prm>
        </Mandatory>
    </Parameters>
</Syntax>
</CounterBoring>
<PeckDrilling>
    <Modal>true</Modal>
    <Syntax>

```

```

<Format>$*G83*$X@axis0$Y@axis1$Z@axis2$R@hight$Q@pechdepth$F@feedrate$K
@repeat</Format>

```

```

    <Parameters>
        <Optional>
            <Prm>$G83</Prm>
            <Prm>$F@FeedRate</Prm>
            <Prm>$K@Repeat</Prm>
        </Optional>
        <Mandatory>
            <Prm>$Q@PechDepth</Prm>
            <Prm>$R@StartPoint</Prm>
            <Prm>$Z@AbsoluteDepth</Prm>
        </Mandatory>
    </Parameters>
</Syntax>
</PeckDrilling>

```

```
<BoringWithStop>
  <Modal>true</Modal>
  <Syntax>

<Format>*$G86*$X@axis0$Y@axis1$Z@axis2$R@hight$F@feedrate$K@repeat</Form
at>
  <Parameters>
    <Optional>
      <Prm>$G86</Prm>
      <Prm>$F@FeedRate</Prm>
      <Prm>$K@Repeat</Prm>
    </Optional>
    <Mandatory>
      <Prm>$R@StartPoint</Prm>
      <Prm>$Z@AbsoluteDepth</Prm>
    </Mandatory>
  </Parameters>
</Syntax>
</BoringWithStop>
<Boring>
  <Modal>true</Modal>
  <Syntax>

<Format>*$G85*$X@axis0$Y@axis1$Z@axis2$R@hight$F@feedrate$K@repeat</Form
at>
  <Parameters>
    <Optional>
      <Prm>$G85</Prm>
      <Prm>$F@FeedRate</Prm>
      <Prm>$K@Repeat</Prm>
    </Optional>
    <Mandatory>
      <Prm>$R@StartPoint</Prm>
      <Prm>$Z@AbsoluteDepth</Prm>
    </Mandatory>
  </Parameters>
</Syntax>
</Boring>
<toolChange>
  <Modal>>false</Modal>
  <Syntax><Format>*$M06*</Format></Syntax>
  <Syntax><Format>*$M6*</Format></Syntax>
</toolChange>
</Movements>

<Settings>
  <ToolFunction>
    <Modal>>false</Modal>
    <Syntax>
      <Format>*$T*$@toolReference</Format>
      <Parameters>
        <Mandatory>
```

```
        <Prm>$T@toolReference</Prm>
        </Mandatory>
        </Parameters>
    </Syntax>
</ToolFunction>
<PlaneXY>
    <Modal>true</Modal>
    <Syntax>
        <Format>$*G17*</Format>
    </Syntax>
</PlaneXY>
<PlaneXZ>
    <Modal>true</Modal>
    <Syntax>
        <Format>$*G18*</Format>
    </Syntax>
</PlaneXZ>
<PlaneYZ>
    <Modal>true</Modal>
    <Syntax>
        <Format>$*G19*</Format>
    </Syntax>
</PlaneYZ>

<SpindleColock>
    <Modal>true</Modal>
    <Syntax>
        <Format>$*M03*</Format>
    </Syntax>
    <Syntax>
        <Format>$*M3*</Format>
    </Syntax>
</SpindleColock>
<SpindleCC>
    <Modal>true</Modal>
    <Syntax>
        <Format>$*M4*</Format>
    </Syntax>
</SpindleCC>
<SpindleOff>
    <Modal>true</Modal>
    <Syntax>
        <Format>$*M5*</Format>
    </Syntax>
</SpindleOff>
<SpindleSpeed>
    <Modal>true</Modal>
    <Syntax>
        <Format>$*S*$@speed</Format>
    </Syntax>
</SpindleSpeed>
<CoolantOn>
```

```
<Modal>true</Modal>
<Syntax>
  <Format>$*M8*</Format>
</Syntax>
</CoolantOn>
<CoolantOff>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*M9*</Format>
  </Syntax>
</CoolantOff>
<FeedRate>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*F*$@rate</Format>
  </Syntax>
</FeedRate>
<ReturnInitial>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G98*</Format>
  </Syntax>
</ReturnInitial>
<ReturnR>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G99*</Format>
  </Syntax>
</ReturnR>
<FeedperMin>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G94*</Format>
  </Syntax>
</FeedperMin>
<FeedperRotation>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G95*</Format>
  </Syntax>
</FeedperRotation>
<CommandMannerABS>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G90*</Format>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$G90</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
```

```
</CommandMannerABS>
<CommandMannerINC>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G91*</Format>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$G91</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
</CommandMannerINC>
<ToolOffsetCancel>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G49*</Format>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$G49</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
</ToolOffsetCancel>
<ToolOffsetPositive>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G43*$H@index</Format>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$G43</Prm>
        <Prm>$H@index</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
</ToolOffsetPositive>
<ToolOffsetNegative>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G44*</Format>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$G44</Prm>
        <Prm>$H@index</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
```

```
</ToolOffsetNegative>
<CutterCompensationCancel>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G40*</Format>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$*G40*</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
</CutterCompensationCancel>
<CutterCompensationLeft>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G41*$H@index</Format>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$G41</Prm>
        <Prm>$H@index</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
</CutterCompensationLeft>
<CutterCompensationRight>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G42*$H@index</Format>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$G42</Prm>
        <Prm>$H@index</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
</CutterCompensationRight>
<UnitINCH>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G70*</Format>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$G70</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
</UnitINCH>
<UnitMM>
```

```
<Modal>true</Modal>
<Syntax>
  <Format>$*G71*</Format>
  <Parameters>
    <Optional/>
    <Mandatory>
      <Prm>$G71</Prm>
    </Mandatory>
  </Parameters>
</Syntax>
</UnitMM>
<SubProgStop>
  <Modal>false</Modal>
  <Syntax>
    <Format>$*M99*</Format>
  </Syntax>
</SubProgStop>
<Stop>
  <Modal>false</Modal>
  <Syntax>
    <Format>$*M00*</Format>
  </Syntax>
  <Syntax>
    <Format>$*M1*</Format>
  </Syntax>
</Stop>
<End>
  <Modal>false</Modal>
  <Syntax>
    <Format>$*M02*</Format>
  </Syntax>
  <Syntax>
    <Format>$*M30*</Format>
  </Syntax>
</End>
</Settings>
</root>
```


B.2. XML specifications of Siemens 840D

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  Document : Siemens.xml
  Created on : 24 May 2010, 11:00
  Author : Xianzhi Zhang
  Description:
    G&M codes programming specification of Siemens 840D controllers.
-->

<root xmlns="http://www.w3schools.com">
  <Make>Siemens</Make>
  <Version>X3</Version>
  <Author>Xianzhi Zhang</Author>
  <Date>2010-05-24</Date>
  <Grammar>
    <Comment>
      <Start>;</Start>
    </Comment>
  </Grammar>
  <Position>
    <Parameters>
      <Mandatory>
        <Prm>$X@axis0</Prm>
        <Prm>$Y@axis1</Prm>
        <Prm>$Z@axis2</Prm>
      </Mandatory>
    </Parameters>
  </Position>
  <Movements>
    <RapidPosition>
      <Modal>>true</Modal>
      <Syntax>
        <Format>$*G0*$X@axis0$Y@axis1$Z@axis2</Format>

        <Parameters>
          <Optional>
            <Prm>$G0</Prm>
          </Optional>
          <Mandatory/>
        </Parameters>
      </Syntax>
    </RapidPosition>
    <LinearInterpolation>
      <Modal>>true</Modal>
      <Syntax>
        <Format>$*G1*$X@axis0$Y@axis1$Z@axis2$F@feedrate</Format>
        <Parameters>
          <Optional>
            <Prm>$G1</Prm>
          </Optional>
        </Parameters>
      </Syntax>
    </LinearInterpolation>
  </Movements>
</root>
```

```

        <Prm>$F@feedrate</Prm>
    </Optional>
    <Mandatory/>
    <Regulations>
        <ParNum>2</ParNum>
    </Regulations>
</Parameters>
</Syntax>
</LinearInterpolation>
<CWCIRCULARInterpolation>
    <Modal>true</Modal>
    <Syntax>

<Format>$*G2*$X@axis0$Y@axis1$Z@axis2$I=AC(@R1)$J=AC(@R2)$K=AC(@R3)$H@
helical$F@feedrate</Format>
    <Parameters>
        <Optional>
            <Prm>$G@02</Prm>
            <Prm>$I@cx</Prm>
            <Prm>$J@cy</Prm>
            <Prm>$K@cz</Prm>
            <Prm>$I=AC(@|cx</Prm>
                <Prm>$J=AC(@|cy</Prm>
                <Prm>$K=AC(@|cz</Prm>
                <Prm>$H@helical</Prm>
                <Prm>$F@feedrate</Prm>
            </Optional>
            <Mandatory/>
            <Regulations>

        </Regulations>
    </Parameters>
</Syntax>
<Syntax>

<Format>$*G2*$X@axis0$Y@axis1$Z@axis2$R@radius$H@helical$F@feedrate</Form
at>
    <Parameters>
        <Optional>
            <Prm>$G02</Prm>
            <Prm>$CR=@radius</Prm>
            <Prm>$H@helical</Prm>
            <Prm>$F@feedrate</Prm>
        </Optional>
        <Mandatory>
            <Prm>$G17/18/19</Prm>
            <Prm>$R@radius</Prm>
        </Mandatory>
        <Regulation>

    <test>null</test>

    </Regulation>
</Parameters>

```

```

        </Syntax>
    </CWCIRCULARInterpolation>

    <CCWCIRCULARInterpolation>
        <Modal>true</Modal>
        <Syntax>

<Format>$$*G3*$X@axis0$Y@axis1$Z@axis2$I=AC(@R1)$J=AC(@R2)$K=AC(@R3)$H
@helical$F@feedrate</Format>
        <Parameters>
            <Optional>
                <Prm>$G03</Prm>
                <Prm>$I@cx</Prm>
                <Prm>$J@cy</Prm>
                <Prm>$K@cz</Prm>
                <Prm>$I=AC(@|cx</Prm>
                <Prm>$J=AC(@|cy</Prm>
                <Prm>$K=AC(@|cz</Prm>
                <Prm>$H@helical</Prm>
                <Prm>$F@feedrate</Prm>
            </Optional>
            <Mandatory/>
        </Parameters>
    </Syntax>
    <Syntax>

<Format>$*G3*$X@axis0$Y@axis1$Z@axis2$R@radius$H@helical$F@feedrate</Form
at>
        <Parameters>
            <Optional>
                <Prm>$G03</Prm>
                <Prm>$R@radius</Prm>
                <Prm>$H@helical</Prm>
                <Prm>$F@feedrate</Prm>
            </Optional>
            <Mandatory>
                <Prm>$R@radius</Prm>
            </Mandatory>
        </Parameters>
    </Syntax>
</CCWCIRCULARInterpolation>
<CycleCancel>
    <Modal>true</Modal>
    <Syntax>
        <Format>$*CYCLE80*</Format>
        <Parameters>
            <Mandatory>
                <Prm>$CYCLE80</Prm>
            </Mandatory>
        </Parameters>
    </Syntax>
</CycleCancel>

```

```

<SpotDrilling>
  <Modal>true</Modal>
  <Syntax>

<Format>*$CYCLE81*($Retr@Retraction$Ref@ReferencePlane$SC@SafetyClearance$FD
@FinalDepth$Rel@RelativeDepth$)</Format>
  <Parameters>
    <Optional>
    </Optional>
    <Mandatory>
    <Prm>$CYCLE81</Prm>
    <Prm>$CYCLE810@Retraction</Prm>
    <Prm>$CYCLE811@ReferencePlane</Prm>
    <Prm>$CYCLE812@SafetyClearance</Prm>
    <Prm>$CYCLE813@FinalDepth</Prm>
    <Prm>$CYCLE814@-RelativeDepth</Prm>
    </Mandatory>
  </Parameters>
</Syntax>
</SpotDrilling>
<CounterBoring>
  <Modal>true</Modal>
  <Syntax>

<Format>*$CYCLE82*($Retr@Retraction$Ref@ReferencePlane$SC@SafetyClearance$FD
@FinalDepth$Rel@RelativeDepth$DT@DwellTime$)</Format>
  <Parameters>
    <Optional>
    <Prm>$CYCLE825@DwellTime</Prm>
    </Optional>
    <Mandatory>
    <Prm>$CYCLE82</Prm>
    <Prm>$CYCLE820@Retraction</Prm>
    <Prm>$CYCLE821@ReferencePlane</Prm>
    <Prm>$CYCLE822@SafetyClearance</Prm>
    <Prm>$CYCLE823@FinalDepth</Prm>
    <Prm>$CYCLE824@-RelativeDepth</Prm>
    </Mandatory>
  </Parameters>
</Syntax>
</CounterBoring>
<PeckDrilling>
  <Modal>>false</Modal>
  <Syntax>

<Format>*$CYCLE83*($Retr@Retraction$Ref@ReferencePlane$SC@SafetyClearance$FD
@FinalDepth$Rel@RelativeDepth$FD@firstdepth$Q@pechdepth$F@feedrate$K@repea
t$)</Format>
  <Parameters>
    <Optional>
    <Prm>$CYCLE83</Prm>
    <Prm>$CYCLE830@Retraction</Prm>

```

```

        <Prm>$CYCLE831@ReferencePlane</Prm>
        <Prm>$CYCLE832@SafetyClearance</Prm>
        <Prm>$CYCLE833@FinalDepth</Prm>
        <Prm>$CYCLE834@-RelativeDepth</Prm>
        <Prm>$CYCLE835@FirstDepth</Prm>
        <Prm>$CYCLE836@PechDepth</Prm>
        <Prm>$CYCLE837@FeedRate</Prm>
        <Prm>$CYCLE838@Repeat</Prm>
            </Optional>
        <Mandatory>

        <Prm>$CYCLE832+CYCLE834@AbsoluteDepth</Prm>
            </Mandatory>
        </Parameters>

    </Syntax>
</PeckDrilling>
<Boring>
    <Modal>>false</Modal>
    <Syntax>

<Format>$*CYCLE85*($Retr@Retraction$Ref@ReferencePlane$SC@SafetyClearance$FD
@FinalDepth$Rel@RelativeDepth$Q@pechdepth$F@feedrate$K@repeat$)</Format>
    <Parameters>

        <Optional>
            <Prm>$CYCLE85</Prm>

            <Prm>$CYCLE850@Retraction</Prm>
            <Prm>$CYCLE851@ReferencePlane</Prm>
            <Prm>$CYCLE852@SafetyClearance</Prm>
            <Prm>$CYCLE853@FinalDepth</Prm>
            <Prm>$CYCLE854@-RelativeDepth</Prm>
            <Prm>$CYCLE855@FirstDepth</Prm>
            <Prm>$CYCLE856@PechDepth</Prm>
            <Prm>$CYCLE857@FeedRate</Prm>
            <Prm>$CYCLE858@Repeat</Prm>
                </Optional>
            <Mandatory>

            <Prm>$CYCLE852+CYCLE854@AbsoluteDepth</Prm>
                </Mandatory>
        </Parameters>

    </Syntax>
</Boring>

<toolChange>
    <Modal>>false</Modal>
    <Syntax><Format>$*M06*$</Format></Syntax>
    <Syntax><Format>$*M6*$</Format></Syntax>
</toolChange>
</Movements>

    <Settings>
    <ToolFunction>

```

```
<Modal>true</Modal>
<Syntax>
  <Format>*$T*@toolReference$</Format>
  <Parameters><Prm>$T@toolReference</Prm></Parameters>
</Syntax>
</ToolFunction>
<Mcall>
  <Modal>false</Modal>
  <Syntax>
    <Format>*$MCALL*$</Format>
  </Syntax>
</Mcall>

<SpindleColock>
  <Modal>true</Modal>
  <Syntax>
    <Format>*$M03*$</Format>
    <Format>*$M3*$</Format>
  </Syntax>
</SpindleColock>
<SpindleCC>
  <Modal>true</Modal>
  <Syntax>
    <Format>*$M04*$</Format>
    <Format>*$M4*$</Format>
  </Syntax>
</SpindleCC>
<SpindleOff>
  <Modal>true</Modal>
  <Syntax>
    <Format>*$M05*$</Format>
    <Format>*$M5*$</Format>
  </Syntax>
</SpindleOff>
<SpindleSpeed>
  <Modal>true</Modal>
  <Syntax>
    <Format>*$S*@speed$</Format>
  </Syntax>
</SpindleSpeed>
<CoolantOn>
  <Modal>true</Modal>
  <Syntax>
    <Format>*$M08*$</Format>
  </Syntax>
</CoolantOn>
<CoolantOff>
  <Modal>true</Modal>
  <Syntax>
    <Format>*$M09*$</Format>
  </Syntax>
</CoolantOff>
```

```

<Spindle>
  <Modal>true</Modal>
  <Syntax>
    <Format>*$S*@speed$</Format>
  </Syntax>
</Spindle>
<FeedRate>
  <Modal>true</Modal>
  <Syntax>
    <Format>*$F*@rate$</Format>
  </Syntax>
</FeedRate>
<ReturnInitial>
  <Modal>true</Modal>
  <Syntax>
    <Format>*$G98*$</Format>
  </Syntax>
</ReturnInitial>
<ReturnR>
  <Modal>true</Modal>
  <Syntax>
    <Format>*$G99*$</Format>
  </Syntax>
</ReturnR>
<feedManner>
  <Modal>true</Modal>
  <Group>05</Group>
  <Syntax>
    <Format>*$G94*$</Format>
    <Format>*$G95*$</Format>
    <Function>perMinute@G94</Function>
    <Function>perRotation@G95</Function>
  </Syntax>
</feedManner>
  <CommandManner>
    <Modal>true</Modal>
    <Syntax>
      <Format>*$G90*</Format>
    <Format>*$G91*</Format>
    <Function>absolute@G90</Function>
    <Function>incremental@G91</Function>
    <Parameter>
      <Optional/>
      <Mandatory>
        <Prm>$G90/91</Prm>
      </Mandatory>
    </Parameter>
  </Syntax>
</CommandManner>
  <ToolOffset>
    <Modal>true</Modal>
    <Syntax>

```

```

    <Format>$*G49*</Format>
    <Function>cancel</Function>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$G49</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
<Syntax>
  <Format>$*G43*$H@index</Format>
  <Function>positive</Function>
  <Parameters>
    <Optional/>
    <Mandatory>
      <Prm>$G43</Prm>
      <Prm>$H@index</Prm>
    </Mandatory>
  </Parameters>
</Syntax>
<Syntax>
  <Format>$*G44*</Format>
  <Function>negative</Function>
  <Parameters>
    <Optional/>
    <Mandatory>
      <Prm>$G44</Prm>
      <Prm>$H@index</Prm>
    </Mandatory>
  </Parameters>
</Syntax>
</ToolOffset>
<CutterCompensation>
  <Modal>true</Modal>
  <Syntax>
    <Format>$*G40*</Format>
    <Function>cancel</Function>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$*G40*</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
  <Syntax>
    <Format>$*G41*$H@index</Format>
    <Function>left</Function>
    <Parameters>
      <Optional/>
      <Mandatory>
        <Prm>$G41</Prm>
        <Prm>$H@index</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>

```

```

        </Mandatory>
      </Parameters>
    </Syntax>
    <Syntax>
      <Format>$*G42*$H@index</Format>
      <Function>right</Function>
      <Parameters>
        <Optional/>
        <Mandatory>
          <Prm>$G42</Prm>
          <Prm>$H@index</Prm>
        </Mandatory>
      </Parameters>
    </Syntax>
  </CutterCompensation>
  <PlaneXY>
    <Modal>true</Modal>
    <Syntax>
      <Format>$*G17*</Format>
    </Syntax>
  </PlaneXY>
  <PlaneXZ>
    <Modal>true</Modal>
    <Syntax>
      <Format>$*G18*</Format>
    </Syntax>
  </PlaneXZ>
  <PlaneYZ>
    <Modal>true</Modal>
    <Syntax>
      <Format>$*G19*</Format>
    </Syntax>
  </PlaneYZ>
  <UnitINCH>
    <Modal>true</Modal>
    <Syntax>
      <Format>$*G70*</Format>
      <Parameters>
        <Optional/>
        <Mandatory>
          <Prm>$G70</Prm>
        </Mandatory>
      </Parameters>
    </Syntax>
  </UnitINCH>
  <UnitMM>
    <Modal>true</Modal>
    <Syntax>
      <Format>$*G71*</Format>
      <Parameters>
        <Optional/>
        <Mandatory>

```

```
        <Prm>$G71</Prm>
      </Mandatory>
    </Parameters>
  </Syntax>
</UnitMM>
<Stop>
  <Modal>>false</Modal>
  <Syntax>
    <Format>$*M00*</Format>
  </Syntax>
</Stop>
<End>
  <Modal>>false</Modal>
  <Syntax>
    <Format>$*M02*</Format>
  </Syntax>
  <Syntax>
    <Format>$*M30*</Format>
  </Syntax>
</End>
</Settings>
</root>
```

Appendix C. Case study results

C.1. Test components and part programmes

C.1.1. Part I

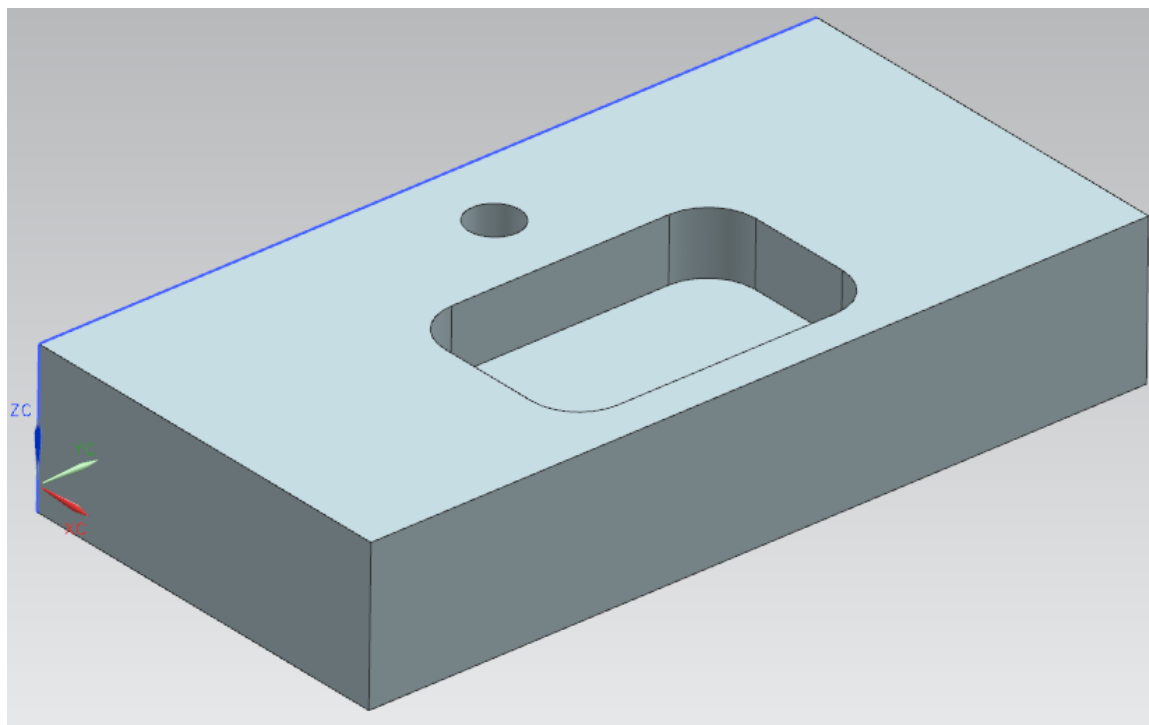


Figure A.1 - 3-D representation – Part I

C.1.1.1. ISO 6983 part programme for Fanuc 18i controller – Part I

%	N245 G0 G17 X26.5 Y36.5	N475 G1 Y36.5 F500
N010 G1902 B50 D100	S4000 M3	N480 X38.5
H15 I0 J0 K0	N250 G43 H10 Z25.0 M8	N485 Y63.5
N020 G17 G90 G94 G57	N255 G8 P1	N490 X26.5
N45 T8 D8 M6	N260 Z2.0	N495 Y55.5
(FACEMILLM6600 66.0)	N265 G1 Z-0.97 F800	N500 X26.127 Y54.462
N50 G0 G17 X90.0 Y20.3	N270 X38.5 Z-1.974	N505 G3 X26.0 Y53.367
S8000 M3	N275 X26.5 Z-2.979	I4.673 J-1.095 F500
N55 G43 H8 Z25.0 M8	N280 X38.5 Z-3.983	N510 G1 Y36.0 F500
N60 G8 P1	N285 X26.5 Z-4.987	N515 X39.0
N65 Z30.0	N290 X38.5 Z-5.991	N520 Y64.0
N70 G1 Z-1.0 F1000.	N295 X26.5 Z-6.996	N525 X26.0
N75 X-33.0	N300 X38.5 Z-8.0	N530 Y50.367
N80 Y73.6	N305 X26.5 F800	N535 G3 X26.127 Y49.271
N85 X83.0	N310 Y43.25	I4.8 J0. F500
N90 G0 Z25.0	N315 X38.5	N540 G1 X26.5 Y48.234
N95 G8 P0	N320 Y50.0	F500
N100 M9	N325 X26.5	N545 G0 Z25.0
N110 M5	N330 Y56.75	N550 M9
N115 (DRILL HOLE1)	N335 X38.5	N1730 M05
N120 G40 G80 G90 G94	N340 Y63.5	N1740 M30
N125 T9 D9 M6	N345 X26.5	%
(TD_M0620:J 6.2)	N350 X33.1	
N135 G0 G17 X10.0 Y50.0	N355 G3 X26.5 Y56.9 I0. J-	
S3000 M3	6.6	
N140 G43 H9 Z25.0 M8	N360 G1 Y36.5	
N145 G83 G98 R2.0 Z-	N365 X38.5	
12.893 Q6.3 F350.	N370 Y63.5	
N150 G0 G80 Z25.0	N375 X26.5	
N155 M9	N380 Y56.9	
N165 M5	N385 G0 Z25.0	
N170 (REAM HOLE1)	N390 (RECTANGULAR	
N175 G40 G80 G90 G94	POCKET FINISH	
N180 T7 D7 M6	RECT_POCK1)	
(REAM_M0660 6.34)	N395 G0 G17 G94 X26.5	
N185 M1	Y56.9 Z25.0 S5000	
N190 G0 G17 X10.0 Y50.0	N400 Y36.5	
S1913 M3	N405 Z2.0	
N195 G43 H7 Z25.0 M8	N410 G1 Z-7.97 F500	
N200 G85 G98 R2.0 Z-11.0	N415 X38.5 Z-9.0	
F1092.	N420 X26.5 F500	
N205 G0 G80 Z25.0	N425 Y43.25	
N210 M9	N430 X38.5	
N220 M5	N435 Y50.0	
N225 (RECTANGULAR	N440 X26.5	
POCKET RECT_POCK1)	N445 Y56.75	
N230 G40 G80 G90 G94	N450 X38.5	
N235 T10 D10 M6	N455 Y63.5	
(ENDMILLM1200:REG	N460 X26.5	
12.0)	N465 X34.5	
N240 M1	N470 G3 X26.5 Y55.5 I0. J-	
	8.0 F500	

C.1.1.2. ISO 6983 part programme for Siemens 840D controller– Part I

;S_ISO_14649_11_RF	N240 M01	N470X38.5
;5-2-2012	N245G94	N475Y63.5
N20G54 SUPA D0	N250S1913M3	N480X26.5
N25G17 G21 G94 G90	N255 G4F4	N485Y56.9
G64 SOFT	N260G0X10.0Y50.0	N490G0Z25.0
N30 ;TOOL	N265Z25.0M8	N495G94
CHANGE(facemillM6600)	N270Z2.0	N500S5000F500
N35 ;FACE FINISH FACE1	N275F1092.	N505Y36.5
N40T1	N280MCALL Cycle85(10,-	N510Z2.0
N45M6	1.0,3.0,,10.0,3,1092.,1000)	N515G1Z-7.97
N50D8	N285X10.0Y50.0	N520X38.5Z-9.0
N55M01	N290MCALL	N525X26.5F500
N60S8000M3	N295Z25.0	N530Y43.25
N65 G4F4	N300Z150M9	N535X38.5
N70G0G54X83.0Y20.3	N305X0Y0M5	N540Y50.0
N75Z25.0M8	N310 ;TOOL	N545X26.5
N80Z3.0	CHANGE(endmillM1200:r	N550Y56.75
N85G1Z-1.0F1000.	eg)	N555X38.5
N90X-33.0	N315 ;RECTANGULAR	N560Y63.5
N95Y73.6	POCKET ROUGH1	N565X26.5
N100X83.0	RECT_POCK1	N570X34.5
N105G0Z25.0	N320T4	N575G3X26.5Y55.5
N110Z150M9	N325M6	I=AC(34.5) J=AC(55.5)
N115X0Y0M5	N330D10	N580G1Y36.5
N120 ;TOOL	N335 M01	N585X38.5
CHANGE(TD_M0620:J)	N340G94	N590Y63.5
N125 ;DRILL HOLE1	N345S4000M3	N595X26.5
N130T2	N350 G4F4	N600Y55.5
N135M6	N355G0X26.5Y36.5	N605X26.127Y54.462
N140D9	N360Z25.0M8	N610G3X26.0Y53.367
N145 M01	N365Z2.0	I=AC(30.8) J=AC(53.367)
N150G94	N370G1Z-0.97F800	N615G1Y36.0
N155S3000M3	N375X38.5Z-1.974	N620X39.0
N160 G4F4	N380X26.5Z-2.979	N625Y64.0
N165G0X10.0Y50.0	N385X38.5Z-3.983	N630X26.0
N170Z25.0M8	N390X26.5Z-4.987	N635Y50.367
N175Z2.0	N395X38.5Z-5.991	N640G3X26.127Y49.271
N180F350	N400X26.5Z-6.996	I=AC(30.8) J=AC(50.367)
N185MCALL Cycle83(10,-	N405X38.5Z-8.0	N645G1X26.5Y48.234
1.0,3.0,,11.893,,6.3,0.8,,,1	N410X26.5F800	N650G0Z25.0
,1,,2)	N415Y43.25	N655Z150M9
N190X10.0Y50.0	N420X38.5	N660X0Y0M5
N195MCALL	N425Y50.0	N665G54 SUPA D0
N200Z25.0	N430X26.5	N670M30
N205Z150M9	N435Y56.75	N675%
N210X0Y0M5	N440X38.5	
N215 ;TOOL	N445Y63.5	
CHANGE(ream_M0634)	N450X26.5	
N220 ;REAM HOLE1	N455X33.1	
N225T3	N460G3X26.5Y56.9	
N230M6	I=AC(33.1) J=AC(56.9)	
N235D7	N465G1Y36.5	

C.1.2. Part II

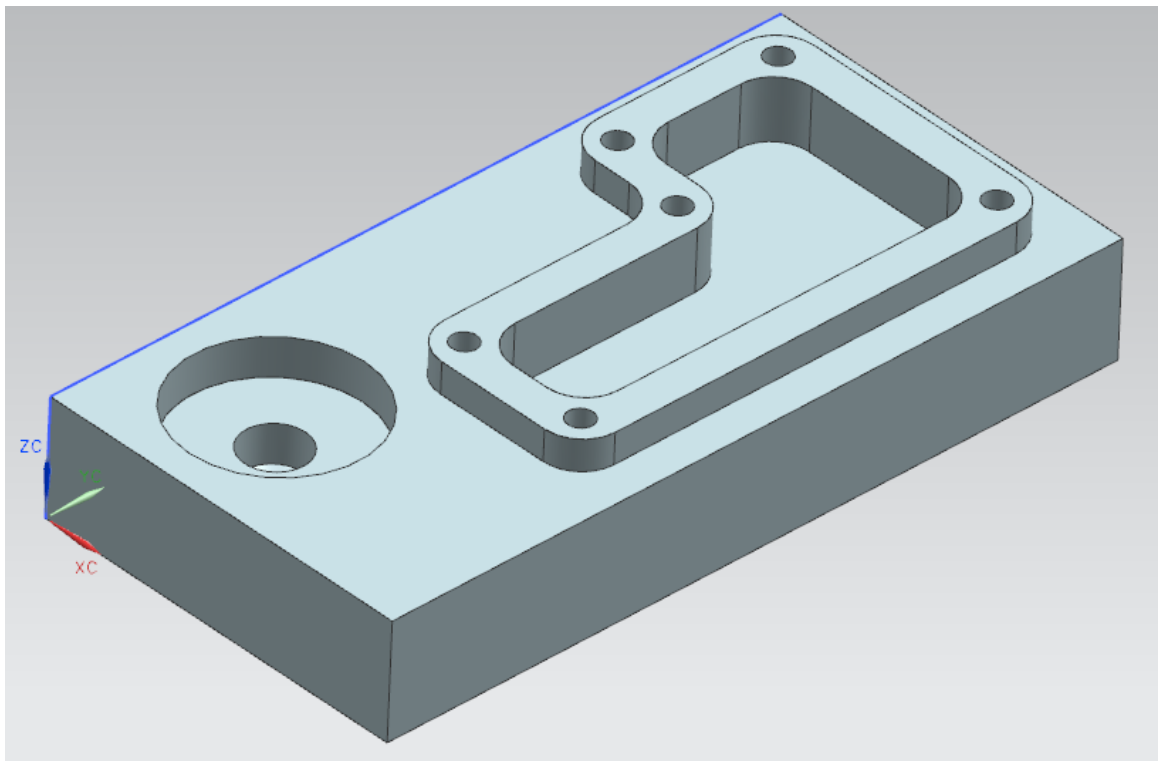


Figure A.2 - 3D representation – Part II

C.1.2.1. ISO 6983 part programme for Fanuc 18i controller– Part II

%	N230 G1 Z-3.75 F631.	N415 G1 X-10.445 Y51.63
N010 G1902 B50 D100	N235 X53.31 Y-3.748	N420 X-10.297 Y56.798
H15 I0 J0 K0	F1261.	N425 X-5.0 Y55.317
N020 G17 G90 G94 G57	N240 G2 X48.862 Y-5.0 I-	N430 G3 X4.01 Y49.883
N35 (FACE FINISH	13.31 J38.748	I15.0 J14.683
FACE1)	N245 G1 X44.807 Y-8.715	N435 G1 Y35.0
N40 G40 G80 G90 G94	N250 X60.464 Y-0.146	N440 G3 X25.0 Y14.01
N45 T8 D8 M6	N255 X55.0 Y0.478	I20.99 J0.
(FACEMILLM6600 66.0)	N260 G2 X40.0 Y-2.64 I-	N445 G1 X40.0
N50 G0 G17 X90.0 Y20.3	15.0 J34.522	N450 G3 X55.0 Y20.317 I0.
S8000 M3	N265 G1 X25.0	J20.99
N55 G43 H8 Z25.0 M8	N270 G2 X-5.0 Y12.267 I0.	N455 G1 X60.297 Y21.798
N60 G8 P1	J37.64	N460 X59.85 Y28.274
N65 Z20.0	N275 G1 X-10.069	N465 X55.0 Y25.679
N70 G1 Z-1.0 F1000.	Y14.403	N470 G2 X40.0 Y17.34 I-
N75 X-33.0	N280 X-9.716 Y21.182	15.0 J9.321
N80 Y73.6	N285 X-5.0 Y18.351	N475 G1 X25.0
N85 X83.0	N290 G3 X25.0 Y0.69	N480 G2 X7.34 Y35.0 I0.
N90 G0 Z25.0	I30.0 J16.649	J17.66
N95 G8 P0	N295 G1 X40.0	N485 G1 Y52.541
N100 M9	N300 G3 X55.0 Y4.143 I0.	N490 G2 X-5.0 Y60.679
N110 M5	J34.31	I2.66 J17.459
N115 (BOSS BOSS2)	N305 G1 X60.486 Y3.752	N495 G1 X-9.85 Y63.274
N120 G40 G80 G90 G94	N310 X60.499 Y7.794	N500 Y96.726
N125 T5 D5 M6	N315 X55.0 Y7.894	N505 X-5.0 Y99.321
(ENDMILLM1000:REG	N320 G2 X40.0 Y4.02 I-	N510 G2 X0.679 Y105.0
10.0)	15.0 J27.106	I15.0 J-9.321
N130 M1	N325 G1 X25.0	N515 G1 X3.274 Y109.85
N135 G0 G17 X1.468 Y-	N330 G2 X-5.0 Y27.27 I0.	N520 G0 Z25.0
9.8 S6306 M3	J30.98	N525 X46.726
N140 G43 H5 Z25.0 M8	N335 G1 X-8.852 Y31.196	N530 Z2.0
N145 G8 P1	N340 X-10.493 Y47.046	N535 G1 Z-3.75 F631.
N150 Z20.0	N345 X-5.0 Y46.772	N540 X49.321 Y105.0
N155 G1 Z-3.75 F631.	N350 G3 X-2.65 Y45.413	F1261.
N160 X-1.019 Y-4.895	I15.0 J23.228	N545 G2 X55.0 Y99.321 I-
F1261.	N355 G1 Y35.0	9.321 J-15.0
N165 G2 X-4.3 Y-2.552	N360 G3 X25.0 Y7.35	N550 G1 X59.85 Y96.726
I26.019 J39.895	I27.65 J0.	N555 G3 X54.329 Y90.131
N170 G1 X-9.747 Y-1.79	N365 G1 X40.0	I1.108 J-6.535
N175 X-10.367 Y3.606	N370 G3 X55.0 Y11.772 I0.	N560 G2 X54.33 Y90.0 I-
N180 X-5.0 Y2.404	J27.65	14.329 J-0.065
N185 G3 X5.961 Y-5.0	N375 G1 X60.493 Y12.046	N565 G1 Y35.0
I30.0 J32.596	N380 X60.445 Y16.63	N570 G2 X40.0 Y20.67 I-
N190 G1 X9.08 Y-9.53	N385 X55.0 Y15.857	14.33 J0.
N195 X20.193 Y-8.715	N390 G2 X40.0 Y10.68 I-	N575 G1 X25.0
N200 X16.138 Y-5.0	15.0 J19.143	N580 G2 X10.67 Y35.0 I0.
N205 G2 X-5.0 Y7.098	N395 G1 X25.0	J14.33
I8.862 J40.0	N400 G2 X0.68 Y35.0 I0.	N585 G1 Y55.67
N210 G1 X-10.258 Y8.713	J24.32	N590 X10.0
N215 G0 Z25.0	N405 G1 Y47.537	N595 G2 X-4.33 Y70.0 I0.
N220 X58.708 Y-4.801	N410 G2 X-5.0 Y50.857	J14.33
N225 Z2.0	I9.32 J22.463	N600 G1 Y90.0

N605 G2 X10.0 Y104.33 I14.33 J0.	N790 G1 X25.0 F1164.	N970 G1 X7.941 Y109.236 F1164.
N610 G1 X40.0	N795 G2 X-5.0 Y15.1 I0. J36.0 F1352.	N975 X42.059
N615 G2 X53.572 Y94.599 I0. J-14.33	N800 G1 X-9.925 Y17.549 F1164.	N980 X45.568 Y105.0
N620 X52.501 Y91.865 I- 2.158 J-0.731	N805 X-8.861 Y31.106	N985 G2 X55.0 Y95.568 I- 5.568 J-15.0 F1693.
N625 G3 X51.0 Y89.341 I1.371 J-2.524	N810 X-5.0 Y27.19	N990 G1 X59.236 Y92.059 F1164.
N630 G1 Y35.0	N815 G3 X25.0 Y4.0 I30.0 J7.81 F1388.	N995 G3 X51.0 Y83.714 I0.11 J-8.345 F728.
N635 G2 X40.0 Y24.0 I- 11.0 J0.	N820 G1 X40.0 F1164.	N1000 G1 Y35.0 F1164.
N640 G1 X25.0	N825 G3 X55.0 Y7.871 I0. J31.0 F1388.	N1005 G2 X40.0 Y24.0 I- 11.0 J0. F1746.
N645 G2 X14.0 Y35.0 I0. J11.0	N830 G1 X60.499 Y7.769 F1164.	N1010 G1 X25.0 F1164.
N650 G1 Y59.0	N835 X60.477 Y14.265	N1015 G2 X14.0 Y35.0 I0. J11.0 F1746.
N655 X10.0	N840 X55.0 Y13.763	N1020 G1 Y59.0 F1164.
N660 G2 X-1.0 Y70.0 I0. J11.0	N845 G2 X40.0 Y9.0 I-15.0 J21.237 F1441.	N1025 X10.0
N665 G1 Y90.0	N850 G1 X25.0 F1164.	N1030 G2 X-1.0 Y70.0 I0. J11.0 F1746.
N670 G2 X10.0 Y101.0 I11.0 J0.	N855 G2 X-1.0 Y35.0 I0. J26.0 F1441.	N1035 G1 Y90.0 F1164.
N675 G1 X40.0	N860 G1 Y46.441 F1164.	N1040 G2 X10.0 Y101.0 I11.0 J0. F1746.
N680 G2 X51.0 Y90.0 I0. J- 11.0	N865 G2 X-5.0 Y48.763 I11.0 J23.558 F1441.	N1045 G1 X40.0 F1164.
N685 G1 Y89.341	N870 G1 X-10.477 Y49.265 F1164.	N1050 G2 X51.0 Y90.0 I0. J-11.0 F1746.
N690 G0 Z25.0	N875 X-10.298 Y56.781	N1055 G1 Y83.714 F1164.
N695 (BOSS FINISH BOSS2)	N880 X-5.0 Y55.303	N1060 X50.263 Y82.766
N700 G0 G17 G94 X51.0 Y89.341 Z25.0 S9702	N885 G3 X4.0 Y49.875 I15.0 J14.697 F1528.	N1065 G3 X50.0 Y81.593 I2.487 J-1.173 F582.
N705 X5.068 Y-9.743	N890 G1 Y35.0 F1164.	N1070 G1 Y35.0 F1164.
N710 Z2.0	N895 G3 X25.0 Y14.0 I21.0 J0. F1528.	N1075 G2 X40.0 Y25.0 I- 10.0 J0. F1746.
N715 G1 Z-5.0 F582.	N900 G1 X40.0 F1164.	N1080 G1 X25.0 F1164.
N720 X2.284 Y-5.0 F1164.	N905 G3 X55.0 Y20.303 I0. J21.0 F1528.	N1085 G2 X15.0 Y35.0 I0. J10.0 F1746.
N725 G2 X-5.0 Y0.129 I22.716 J40.0 F1306.	N910 G1 X60.298 Y21.781 F1164.	N1090 G1 Y60.0 F1164.
N730 G1 X-10.404 Y1.151 F1164.	N915 X59.236 Y32.941	N1095 X10.0
N735 X-10.259 Y8.664	N920 X55.0 Y29.432	N1100 G2 X0. Y70.0 I0. J10.0 F1746.
N740 X-5.0 Y7.054	N925 G2 X40.0 Y19.0 I- 15.0 J5.568 F1693.	N1105 G1 Y90.0 F1164.
N745 G3 X16.0 Y-5.0 I30.0 J27.946 F1326.	N930 G1 X25.0 F1164.	N1110 G2 X10.0 Y100.0 I10.0 J0. F1746.
N750 G1 X20.043 Y-8.728 F1164.	N935 G2 X9.0 Y35.0 I0. J16.0 F1693.	N1115 G1 X40.0 F1164.
N755 X44.957	N940 G1 Y54.031 F1164.	N1120 G2 X50.0 Y90.0 I0. J-10.0 F1746.
N760 X49.0 Y-5.0	N945 G2 X-5.0 Y64.432 I1.0 J15.969 F1693.	N1125 G1 Y78.593 F1164.
N765 G3 X53.249 Y-3.8 I- 9.0 J40.0 F1326.	N950 G1 X-9.236 Y67.941 F1164.	N1130 G3 X50.263 Y77.421 I2.75 J0. F582.
N770 G1 X58.646 Y-4.863 F1164.	N955 Y92.059	N1135 G1 X51.0 Y76.472 F1164.
N775 X60.476 Y1.759	N960 X-5.0 Y95.568	N1140 G0 Z25.0
N780 X55.0 Y2.274	N965 G2 X4.432 Y105.0 I15.0 J-5.568 F1693.	N1145 M9
N785 G2 X40.0 Y-1.0 I- 15.0 J32.726 F1352.		N1155 M5

N1160 (POCKET POCKET1)	N1375 G0 G17 G94 X23.233 Y85.0 Z25.0 S9000	N1575 (POCKET ROUGH1 HOLE3)
N1165 G40 G80 G90 G94	N1380 X23.0 Y81.0	N1580 G0 G17 G94 X16.0 Y16.0 Z25.0 S7885
N1170 T4 D4 M6 (ENDMILLM0800:REG 8.0)	N1385 Z2.0	N1585 Z-2.0
N1175 M1	N1390 G1 Z-7.97 F432.	N1590 G1 Z-4.97 F631.
N1180 G0 G17 X25.672 Y79.672 S7882 M3	N1395 X31.0 Z-8.313	N1595 X20.0
N1185 G43 H4 Z25.0 M8	N1400 X23.0 Z-8.657	N1600 G3 X20.0 Y16.0 Z- 5.976 I-4.0 J0.
N1190 G8 P1	N1405 X31.0 Z-9.0	N1605 X20.0 Y16.0 Z- 6.982 I-4.0 J0.
N1195 Z20.0	N1410 X19.0 F864.	N1610 X20.0 Y16.0 Z- 7.988 I-4.0 J0.
N1200 G1 Z-0.97 F631.	N1415 Y79.0	N1615 X20.0 Y16.0 Z- 8.994 I-4.0 J0.
N1205 X33.672 Z-1.609	N1420 X20.0	N1620 X12.0 Y16.0 Z- 9.497 I-4.0 J0.
N1210 X25.672 Z-2.248	N1425 G2 X31.0 Y73.66 I0. J-14.0 F1210.	N1625 X20.0 Y16.0 Z-10.0 I4.0 J0.
N1215 X33.672 Z-2.887	N1430 G1 Y81.0 F864.	N1630 G1 X21.336 F1262.
N1220 X25.672 Z-3.526	N1435 G3 X28.364 Y84.435 I-3.556 J0. F432.	N1635 G3 X21.336 Y16.0 I-5.336 J0.
N1225 X33.672 Z-4.165	N1440 X24.072 Y85.0 I- 4.293 J-16.02 F696.	N1640 G1 X23.117 Y17.525
N1230 X25.672 Z-4.805	N1445 G1 X15.0 F864.	N1645 G3 X23.003 Y19.868 I-2.446 J1.054
N1235 X33.672 Z-5.444	N1450 Y75.0	N1650 X23.003 Y19.868 I- 7.003 J-3.868
N1240 X25.672 Z-6.083	N1455 X20.0	N1655 X21.828 Y21.48 I- 7.003 J-3.868
N1245 X33.672 Z-6.722	N1460 G2 X30.0 Y65.0 I0. J-10.0 F1296.	N1660 X19.634 Y22.307 I- 1.941 J-1.825
N1250 X25.672 Z-7.361	N1465 G1 Y40.0 F864.	N1665 G1 X17.636 Y21.079
N1255 X33.672 Z-8.0	N1470 X35.0	N1670 G0 Z25.0
N1260 X24.437 F1261.	N1475 Y85.0	N1675 M9
N1265 G2 X29.672 Y76.891 I-4.437 J-14.672	N1480 X24.072	N1685 M5
N1270 G1 Y79.672	N1485 X23.277 Y85.734	N1690 (DRILL HOLE4)
N1275 G3 X27.917 Y81.96 I-2.368 J0.	N1490 G3 X22.228 Y86.0 I-1.049 J-1.934 F432.	N1695 G40 G80 G90 G94
N1280 X25.058 Y82.336 I- 2.859 J-10.669	N1495 G1 X15.0 F864.	N1700 T7 D7 M6 (TD_M0830:J 8.3)
N1285 G1 X17.664	N1500 G3 X14.0 Y85.0 I0. J-1.0 F432.	N1710 G0 G17 X16.0 Y16.0 S3031 M3
N1290 Y77.664	N1505 G1 Y75.0 F864.	N1715 G43 H7 Z25.0 M8
N1295 X20.0	N1510 G3 X15.0 Y74.0 I1.0 J0. F432.	N1720 G81 G98 R-7.0 Z- 17.403 F364.
N1300 G2 X32.336 Y67.864 I0. J-12.664	N1515 G1 X20.0 F864.	N1725 G0 G80 Z25.0
N1305 G1 Y82.336	N1520 G2 X29.0 Y65.0 I0. J-9.0 F1296.	N1730 M9
N1310 X27.848	N1525 G1 Y40.0 F864.	N1740 M5
N1315 G2 X25.54 Y83.668 I0. J2.664	N1530 G3 X30.0 Y39.0 I1.0 J0. F432.	N1745 (DRILL HOLE1)
N1320 G3 X23.233 Y85.0 I-2.307 J-1.332	N1535 G1 X35.0 F864.	N1750 G40 G80 G90 G94
N1325 G1 X15.0	N1540 G3 X36.0 Y40.0 I0. J1.0 F432.	N1755 T22 D22 M6 (TD_M0340:J 3.4)
N1330 Y75.0	N1545 G1 Y85.0 F864.	
N1335 X20.0	N1550 G3 X35.0 Y86.0 I- 1.0 J0. F432.	
N1340 G2 X30.0 Y65.0 I0. J-10.0	N1555 G1 X19.228 F864.	
N1345 G1 Y40.0	N1560 G3 X18.179 Y85.734 I0. J-2.2 F432.	
N1350 X35.0	N1565 G1 X17.384 Y85.0 F864.	
N1355 Y85.0	N1570 G0 Z25.0	
N1360 X23.233		
N1365 G0 Z25.0		
N1370 (POCKET FINISH POCKET1)		

N1765 G0 G17 X41.0
 Y91.0 S8085 M3
 N1770 G43 H22 Z25.0 M8
 N1775 G83 G98 R3.0 Z-
 10.901 Q3.0 F364.
 N1780 G0 G80 Z25.0
 N1785 X9.0
 N1790 G83 G98 R3.0 Z-
 10.901 Q3.0 F364.
 N1795 G0 G80 Z25.0
 N1800 Y69.0 N1805 G83
 G98 R3.0 Z-10.901 Q3.0
 F364.
 N1810 G0 G80 Z25.0
 N1815 X21.0 Y66.0
 N1820 G83 G98 R3.0 Z-
 10.901 Q3.0 F364.
 N1825 G0 G80 Z25.0
 N1830 X24.0 Y34.0
 N1835 G83 G98 R3.0 Z-
 10.901 Q3.0 F364.
 N1840 G0 G80 Z25.0
 N1845 X41.0
 N1850 G83 G98 R3.0 Z-
 10.901 Q3.0 F364.
 N1855 G0 G80 Z25.0
 N1860 M9
 N1885 M30
 %

C.1.2.2. ISO 6983 part programme for Siemens 840D controller – Part II

;S_Test2_RF	N230G2X-5.0Y7.098	N425G2X0.68Y35.0
;5-2-2012	I=AC(25.0) J=AC(35.0)	I=AC(25.0) J=AC(35.0)
N20G54 SUPA D0	N235G1X-10.258Y8.713	N430G1Y47.537
N25G17 G21 G94 G90	N240G0Z25.0	N435G2X-5.0Y50.857
G64 SOFT	N245X58.708Y-4.801	I=AC(10.0) J=AC(70.0)
N30 ;TOOL	N250Z2.0	N440G1X-10.445Y51.63
CHANGE(facemillM6600)	N255G1Z-3.75F631.	N445X-10.297Y56.798
N35 ;FACE FINISH FACE1	N260X53.31Y-3.748F1261.	N450X-5.0Y55.317
N40T1	N265G2X48.862Y-5.0	N455G3X4.01Y49.883
N45M6	I=AC(40.0) J=AC(35.0)	I=AC(10.0) J=AC(70.0)
N50D1	N270G1X44.807Y-8.715	N460G1Y35.0
N55M01	N275X60.464Y-0.146	N465G3X25.0Y14.01
N60S8000M3	N280X55.0Y0.478	I=AC(25.0) J=AC(35.0)
N65 G4F4	N285G2X40.0Y-2.64	N470G1X40.0
N70G0G54X83.0Y20.3	I=AC(40.0) J=AC(35.0)	N475G3X55.0Y20.317
N75Z25.0M8	N290G1X25.0	I=AC(40.0) J=AC(35.0)
N80Z3.0	N295G2X-5.0Y12.267	N480G1X60.297Y21.798
N85G1Z-1.0F1000.0	I=AC(25.0) J=AC(35.0)	N485X59.85Y28.274
N90X-33.0	N300G1X-10.069Y14.403	N490X55.0Y25.679
N95Y73.6	N305X-9.716Y21.182	N495G2X40.0Y17.34
N100X83.0	N310X-5.0Y18.351	I=AC(40.0) J=AC(35.0)
N105G0Z25.0	N315G3X25.0Y0.69	N500G1X25.0
N110Z150M9	I=AC(25.0) J=AC(35.0)	N505G2X7.34Y35.0
N115X0Y0M5	N320G1X40.0	I=AC(25.0) J=AC(35.0)
N120 ;TOOL	N325G3X55.0Y4.143	N510G1Y52.541
CHANGE(endmillM1000:reg)	I=AC(40.0) J=AC(35.0)	N515G2X-5.0Y60.679
N125 ;BOSS ROUGH1	N330G1X60.486Y3.752	I=AC(10.0) J=AC(70.0)
BOSS2	N335X60.499Y7.794	N520G1X-9.85Y63.274
N130T2	N340X55.0Y7.894	N525Y96.726
N135M6	N345G2X40.0Y4.02	N530X-5.0Y99.321
N140D2	I=AC(40.0) J=AC(35.0)	N535G2X0.679Y105.0
N145 M01	N350G1X25.0	I=AC(10.0) J=AC(90.0)
N150G94	N355G2X-5.0Y27.27	N540G1X3.274Y109.85
N155S6306M3	I=AC(25.0) J=AC(35.0)	N545G0Z25.0
N160 G4F4	N360G1X-8.852Y31.196	N550X46.726
N165G0X1.468Y-9.8	N365X-10.493Y47.046	N555Z2.0
N170Z25.0M8	N370X-5.0Y46.772	N560G1Z-3.75F631.
N175Z2.0	N375G3X-2.65Y45.413	N565X49.321Y105.0F1261.
N180G1Z-3.75F631.	I=AC(10.0) J=AC(70.0)	N570G2X55.0Y99.321
N185X-1.019Y-4.895F1261.	N380G1Y35.0	I=AC(40.0) J=AC(90.0)
N190G2X-4.3Y-2.552	N385G3X25.0Y7.35	N575G1X59.85Y96.726
I=AC(25.0) J=AC(35.0)	I=AC(25.0) J=AC(35.0)	N580G3X54.329Y90.191
N195G1X-9.747Y-1.79	N390G1X40.0	I=AC(60.958) J=AC(90.191)
N200X-10.367Y3.606	N395G3X55.0Y11.772	N585G1Y90.131
N205X-5.0Y2.404	I=AC(40.0) J=AC(35.0)	N590X54.33Y90.0
N210G3X5.961Y-5.0	N400G1X60.493Y12.046	N595Y35.0
I=AC(25.0) J=AC(35.0)	N405X60.445Y16.63	N600G2X40.0Y20.67
N215G1X9.08Y-9.53	N410X55.0Y15.857	I=AC(40.0) J=AC(35.0)
N220X20.193Y-8.715	N415G2X40.0Y10.68	N605G1X25.0
N225X16.138Y-5.0	I=AC(40.0) J=AC(35.0)	N610G2X10.67Y35.0
	N420G1X25.0	I=AC(25.0) J=AC(35.0)
		N615G1Y55.67

N620X10.0	N820G2X40.0Y-1.0	N1020G2X55.0Y95.568
N625G2X-4.33Y70.0	I=AC(40.0) J=AC(35.0)	I=AC(40.0) J=AC(90.0)
I=AC(10.0) J=AC(70.0)	N825G1X25.0	N1025G1X59.236Y92.059
N630G1Y90.0	N830G2X-5.0Y15.1	N1030G3X51.0Y83.714
N635G2X10.0Y104.33	I=AC(25.0) J=AC(35.0)	I=AC(59.346) J=AC(83.714)
I=AC(10.0) J=AC(90.0)	N835G1X-9.925Y17.549	N1035G1Y35.0
N640G1X40.0	N840X-8.861Y31.106	N1040G2X40.0Y24.0
N645G2X53.572Y94.599	N845X-5.0Y27.19	I=AC(40.0) J=AC(35.0)
I=AC(40.0) J=AC(90.0)	N850G3X25.0Y4.0	N1045G1X25.0
N650X53.692Y93.868	I=AC(25.0) J=AC(35.0)	N1050G2X14.0Y35.0
I=AC(51.413) J=AC(93.868)	N855G1X40.0	I=AC(25.0) J=AC(35.0)
N655X52.501Y91.865	N860G3X55.0Y7.871	N1055G1Y59.0
I=AC(51.413) J=AC(93.868)	I=AC(40.0) J=AC(35.0)	N1060X10.0
N660G3X51.0Y89.341	N865G1X60.499Y7.769	N1065G2X-1.0Y70.0
I=AC(53.872) J=AC(89.341)	N870X60.477Y14.265	I=AC(10.0) J=AC(70.0)
N665G1Y35.0	N875X55.0Y13.763	N1070G1Y90.0
N670G2X40.0Y24.0	N880G2X40.0Y9.0	N1075G2X10.0Y101.0
I=AC(40.0) J=AC(35.0)	I=AC(40.0) J=AC(35.0)	I=AC(10.0) J=AC(90.0)
N675G1X25.0	N885G1X25.0	N1080G1X40.0
N680G2X14.0Y35.0	N890G2X-1.0Y35.0	N1085G2X51.0Y90.0
I=AC(25.0) J=AC(35.0)	I=AC(25.0) J=AC(35.0)	I=AC(40.0) J=AC(90.0)
N685G1Y59.0	N895G1Y46.441	N1090G1Y83.714
N690X10.0	N900G2X-5.0Y48.763	N1095X50.263Y82.766
N695G2X-1.0Y70.0	I=AC(10.0) J=AC(70.0)	N1100G3X50.0Y81.593
I=AC(10.0) J=AC(70.0)	N905G1X-10.477Y49.265	I=AC(52.75) J=AC(81.593)
N700G1Y90.0	N910X-10.298Y56.781	N1105G1Y35.0
N705G2X10.0Y101.0	N915X-5.0Y55.303	N1110G2X40.0Y25.0
I=AC(10.0) J=AC(90.0)	N920G3X4.0Y49.875	I=AC(40.0) J=AC(35.0)
N710G1X40.0	I=AC(10.0) J=AC(70.0)	N1115G1X25.0
N715G2X51.0Y90.0	N925G1Y35.0	N1120G2X15.0Y35.0
I=AC(40.0) J=AC(90.0)	N930G3X25.0Y14.0	I=AC(25.0) J=AC(35.0)
N720G1Y89.341	I=AC(25.0) J=AC(35.0)	N1125G1Y60.0
N725G0Z25.0	N935G1X40.0	N1130X10.0
N730G94	N940G3X55.0Y20.303	N1135G2X0.Y70.0
N735S9702F582.	I=AC(40.0) J=AC(35.0)	I=AC(10.0) J=AC(70.0)
N740X5.068Y-9.743	N945G1X60.298Y21.781	N1140G1Y90.0
N745Z2.0	N950X59.236Y32.941	N1145G2X10.0Y100.0
N750G1Z-5.0	N955X55.0Y29.432	I=AC(10.0) J=AC(90.0)
N755X2.284Y-5.0F1164.	N960G2X40.0Y19.0	N1150G1X40.0
N760G2X-5.0Y0.129	I=AC(40.0) J=AC(35.0)	N1155G2X50.0Y90.0
I=AC(25.0) J=AC(35.0)	N965G1X25.0	I=AC(40.0) J=AC(90.0)
N765G1X-10.404Y1.151	N970G2X9.0Y35.0	N1160G1Y78.593
N770X-10.259Y8.664	I=AC(25.0) J=AC(35.0)	N1165G3X50.263Y77.421
N775X-5.0Y7.054	N975G1Y54.031	I=AC(52.75) J=AC(78.593)
N780G3X16.0Y-5.0	N980G2X-5.0Y64.432	N1170G1X51.0Y76.472
I=AC(25.0) J=AC(35.0)	I=AC(10.0) J=AC(70.0)	N1175G0Z25.0
N785G1X20.043Y-8.728	N985G1X-9.236Y67.941	N1180Z150M9
N790X44.957	N990Y92.059	N1185X0Y0M5
N795X49.0Y-5.0	N995X-5.0Y95.568	N1190 ;TOOL
N800G3X53.249Y-3.8	N1000G2X4.432Y105.0	CHANGE(endmillM0800:r
I=AC(40.0) J=AC(35.0)	I=AC(10.0) J=AC(90.0)	eg)
N805G1X58.646Y-4.863	N1005G1X7.941Y109.236	N1195 ;POCKET ROUGH1
N810X60.476Y1.759	N1010X42.059	POCKET1
N815X55.0Y2.274	N1015X45.568Y105.0	N1200T3
		N1205M6

N1210D3	N1450X23.0Z-8.657	N1650G3X16.0Y20.0Z-
N1215 M01	N1455X31.0Z-9.0	5.221 I=AC(16.0)
N1220G94	N1460X19.0F864.	J=AC(16.0)
N1225S7882M3	N1465Y79.0	N1655X12.0Y16.0Z-5.473
N1230 G4F4	N1470X20.0	I=AC(16.0) J=AC(16.0)
N1235G0X25.672Y79.672	N1475G2X31.0Y73.66	N1660X16.0Y12.0Z-5.724
N1240Z25.0M8	I=AC(20.0) J=AC(65.0)	I=AC(16.0) J=AC(16.0)
N1245Z2.0	N1480G1Y81.0	N1665X20.0Y16.0Z-5.976
N1250G1Z-0.97F631.	N1485G3X28.364Y84.435	I=AC(16.0) J=AC(16.0)
N1255X33.672Z-1.609	I=AC(27.444) J=AC(81.0)	N1670X16.0Y20.0Z-6.228
N1260X25.672Z-2.248	N1490X24.072Y85.0	I=AC(16.0) J=AC(16.0)
N1265X33.672Z-2.887	I=AC(24.072) J=AC(68.415)	N1675X12.0Y16.0Z-6.479
N1270X25.672Z-3.526	N1495G1X15.0	I=AC(16.0) J=AC(16.0)
N1275X33.672Z-4.165	N1500Y75.0	N1680X16.0Y12.0Z-6.73
N1280X25.672Z-4.805	N1505X20.0	I=AC(16.0) J=AC(16.0)
N1285X33.672Z-5.444	N1510G2X30.0Y65.0	N1685X20.0Y16.0Z-6.982
N1290X25.672Z-6.083	I=AC(20.0) J=AC(65.0)	I=AC(16.0) J=AC(16.0)
N1295X33.672Z-6.722	N1515G1Y40.0	N1690X16.0Y20.0Z-7.233
N1300X25.672Z-7.361	N1520X35.0	I=AC(16.0) J=AC(16.0)
N1305X33.672Z-8.0	N1525Y85.0	N1695X12.0Y16.0Z-7.485
N1310X24.437F1261.	N1530X24.072	I=AC(16.0) J=AC(16.0)
N1315G2X29.672Y76.891	N1535X23.277Y85.734	N1700X16.0Y12.0Z-7.736
I=AC(20.0) J=AC(65.0)	N1540G3X22.228Y86.0	I=AC(16.0) J=AC(16.0)
N1320G1Y79.672	I=AC(22.228) J=AC(83.8)	N1705X20.0Y16.0Z-7.988
N1325G3X27.917Y81.96	N1545G1X15.0	I=AC(16.0) J=AC(16.0)
I=AC(27.304) J=AC(79.672)	N1550G3X14.0Y85.0	N1710X16.0Y20.0Z-8.239
N1330X25.058Y82.336	I=AC(15.0) J=AC(85.0)	I=AC(16.0) J=AC(16.0)
I=AC(25.058) J=AC(71.29)	N1555G1Y75.0	N1715X12.0Y16.0Z-8.491
N1335G1X17.664	N1560G3X15.0Y74.0	I=AC(16.0) J=AC(16.0)
N1340Y77.664	I=AC(15.0) J=AC(75.0)	N1720X16.0Y12.0Z-8.742
N1345X20.0	N1565G1X20.0	I=AC(16.0) J=AC(16.0)
N1350G2X32.336Y67.864	N1570G2X29.0Y65.0	N1725X20.0Y16.0Z-8.994
I=AC(20.0) J=AC(65.0)	I=AC(20.0) J=AC(65.0)	I=AC(16.0) J=AC(16.0)
N1355G1Y82.336	N1575G1Y40.0	N1730X16.0Y20.0Z-9.245
N1360X27.848	N1580G3X30.0Y39.0	I=AC(16.0) J=AC(16.0)
N1365G2X25.54Y83.668	I=AC(30.0) J=AC(40.0)	N1735X12.0Y16.0Z-9.497
I=AC(27.848) J=AC(85.0)	N1585G1X35.0	I=AC(16.0) J=AC(16.0)
N1370G3X23.233Y85.0	N1590G3X36.0Y40.0	N1740X16.0Y12.0Z-9.748
I=AC(23.233) J=AC(82.336)	I=AC(35.0) J=AC(40.0)	I=AC(16.0) J=AC(16.0)
N1375G1X15.0	N1595G1Y85.0	N1745X20.0Y16.0Z-10.0
N1380Y75.0	N1600G3X35.0Y86.0	I=AC(16.0) J=AC(16.0)
N1385X20.0	I=AC(35.0) J=AC(85.0)	N1750G1X21.336F1262.
N1390G2X30.0Y65.0	N1605G1X19.228	N1755G3X16.0Y21.336
I=AC(20.0) J=AC(65.0)	N1610G3X18.179Y85.734	I=AC(16.0) J=AC(16.0)
N1395G1Y40.0	I=AC(19.228) J=AC(83.8)	N1760X10.664Y16.0
N1400X35.0	N1615G1X17.384Y85.0	I=AC(16.0) J=AC(16.0)
N1405Y85.0	N1620G0Z25.0	N1765X16.0Y10.664
N1410X23.233	N1625G94	I=AC(16.0) J=AC(16.0)
N1415G0Z25.0	N1630X16.0Y16.0S7885F6	N1770X21.336Y16.0
N1420G94	31.	I=AC(16.0) J=AC(16.0)
N1425S9000F432.	N1635Z-2.0	N1775G1X23.117Y17.525
N1430X23.0Y81.0	N1640G1Z-4.97	N1780G3X23.335Y18.58
N1435Z2.0	N1645X20.0	I=AC(20.671) J=AC(18.58)
N1440G1Z-7.97		N1785X23.003Y19.868
N1445X31.0Z-8.313		I=AC(20.671) J=AC(18.58)

N1790X16.0Y24.0	N2010MCALL	N2225%
I=AC(16.0) J=AC(16.0)	Cycle83(10,0.,3.0,,10.901,	
N1795X8.0Y16.0	,3.0,0.8,,,1,1,,2)	
I=AC(16.0) J=AC(16.0)	N2015X41.0Y91.0	
N1800X16.0Y8.0	N2020MCALL	
I=AC(16.0) J=AC(16.0)	N2025Z25.0	
N1805X24.0Y16.0	N2030X9.0	
I=AC(16.0) J=AC(16.0)	N2035Z3.0	
N1810X23.003Y19.868	N2040F364.	
I=AC(16.0) J=AC(16.0)	N2045MCALL	
N1815X21.828Y21.48	Cycle83(10,0.,3.0,,10.901,	
I=AC(16.0) J=AC(16.0)	,3.0,0.8,,,1,1,,2)	
N1820X19.888Y22.319	N2050X9.0Y91.0	
I=AC(19.888) J=AC(19.655)	N2055MCALL	
N1825X19.634Y22.307	N2060Z25.0	
I=AC(19.888) J=AC(19.655)	N2065Y69.0	
N1830G1X17.636Y21.079	N2070Z3.0	
N1835G0Z25.0	N2075F364.	
N1840Z150M9	N2080MCALL	
N1845X0Y0M5	Cycle83(10,0.,3.0,,10.901,	
N1850 ;TOOL	,3.0,0.8,,,1,1,,2)	
CHANGE(TD_M0830:J)	N2085X9.0Y69.0	
N1855 ;DRILL HOLE4	N2090MCALL	
N1860T4	N2095Z25.0	
N1865M6	N2100X21.0Y66.0	
N1870D4	N2105Z3.0	
N1875 M01	N2110F364.	
N1880G94	N2115MCALL	
N1885S3031M3	Cycle83(10,0.,3.0,,10.901,	
N1890 G4F4	,3.0,0.8,,,1,1,,2)	
N1895G0X16.0Y16.0	N2120X21.0Y66.0	
N1900Z25.0M8	N2125MCALL	
N1905Z-7.0	N2130Z25.0	
N1910F364.	N2135X24.0Y34.0	
N1915MCALL Cycle81(10,-	N2140Z3.0	
10.0,3.0,,7.403)	N2145F364.	
N1920X16.0Y16.0	N2150MCALL	
N1925MCALL	Cycle83(10,0.,3.0,,10.901,	
N1930Z25.0	,3.0,0.8,,,1,1,,2)	
N1935Z150M9	N2155X24.0Y34.0	
N1940X0Y0M5	N2160MCALL	
N1945 ;TOOL	N2165Z25.0	
CHANGE(TD_M0340:J)	N2170X41.0	
N1950 ;DRILL HOLE1	N2175Z3.0	
N1955T5	N2180F364.	
N1960M6	N2185MCALL	
N1965D5	Cycle83(10,0.,3.0,,10.901,	
N1970 M01	,3.0,0.8,,,1,1,,2)	
N1975G94	N2190X41.0Y34.0	
N1980S8085M3	N2195MCALL	
N1985 G4F4	N2200Z25.0	
N1990G0X41.0Y91.0	N2205Z150M9	
N1995Z25.0M8	N2210X0Y0M5	
N2000Z3.0	N2215G54 SUPA D0	
N2005F364.	N2220M30	

C.1.3. Part III

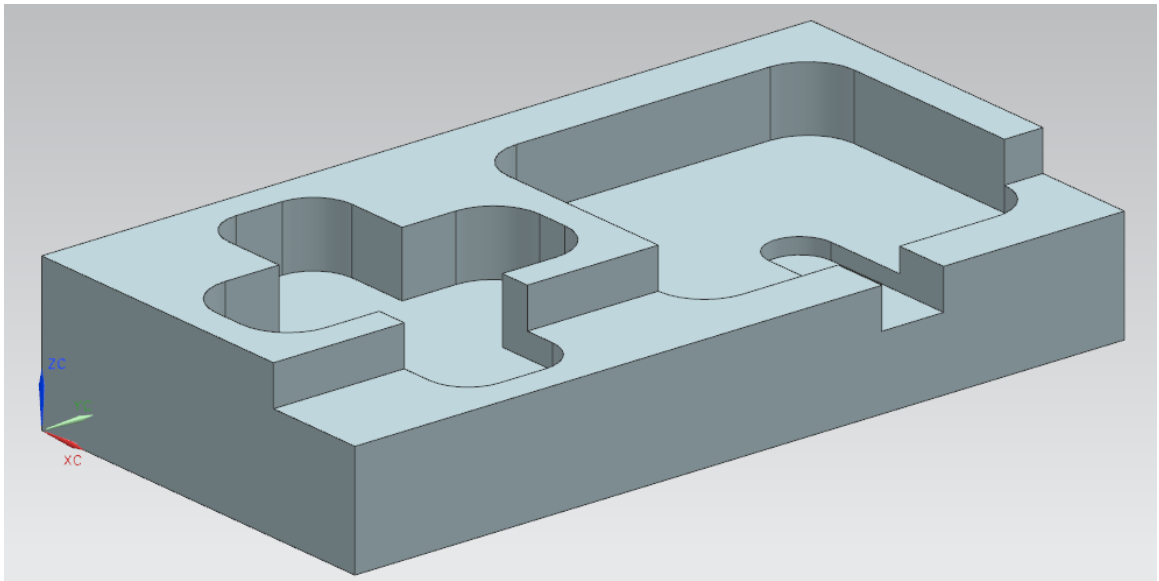


Figure A.3 3-D representation – Part III

C.1.3.1. ISO 6983 part programme for Fanuc 18i controller – Part III

%	N215 X25.0 Y13.3 Z-6.638	N395 X26.905 Y31.01 Z-
N010 G1902 B50 D100	N220 X23.045 Y18.973 Z-	8.055
H15 I0 J0 K0	7.111	N400 X25.0 Y36.7 Z-8.528
N020 G17 G90 G94 G57	N225 X25.0 Y13.3 Z-7.583	N405 X26.905 Y31.01 Z-
N35 (FACE FINISH	N230 X23.045 Y18.973 Z-	9.0
FACE1)	8.055	N410 X25.0 Y36.7 F1262.
N40 G40 G80 G90 G94	N235 X25.0 Y13.3 Z-8.528	N415 X23.545 Y37.917
N45 T8 D8 M6	N240 X23.045 Y18.973 Z-	N420 G2 X23.0 Y39.733
(FACEMILLM6600 66.0)	9.0	I2.755 J1.817
N50 G0 G17 X90.0 Y20.3	N245 X25.0 Y13.3 F1261.	N425 G1 Y40.0
S8000 M3	N250 X26.455 Y14.517	N430 X27.0
N55 G43 H8 Z25.0 M8	N255 G3 X27.0 Y16.333 I-	N435 Y31.0
N60 G8 P1	2.755 J1.817	N440 X23.0
N65 Z20.0	N260 G1 Y19.0	N445 Y40.0
N70 G1 Z-1.0 F1000.	N265 X23.0	N450 X25.733
N75 X-33.0	N270 Y10.0	N455 G2 X26.187 Y39.864
N80 Y73.6	N275 X27.0	I0. J-0.825
N85 X83.0	N280 Y19.0	N460 G1 X26.491 Y39.5
N90 G0 Z25.0	N285 X26.667	N465 G0 Z25.0
N95 G8 P0	N290 G3 X24.85 Y18.455	N470 M9
N100 M9	I0. J-3.3	N480 M5
N110 M5	N295 G1 X23.634 Y17.0	N485 (RECTANGULAR
N115 (RECTANGULAR	N300 G0 Z25.0	POCKET RECT_POCK2)
POCKET RECT_POCK4)		N490 G40 G80 G90 G94
N120 G40 G80 G90 G94	N305 (RECTANGULAR	N495 T5 D5 M6
N125 T10 D10 M6	POCKET ROUGH1	(ENDMILLM1000:REG
(ENDMILLM1200:REG	RECT_POCK5)	10.0)
12.0)	N310 G0 G17 G94 X25.0	N500 M1
N130 M1	Y36.7 Z25.0 S5256	N505 G0 G17 X19.5 Y22.5
N135 G0 G17 X25.0 Y13.3	N315 Z2.0	S6306 M3
S5255 M3	N320 G1 Z-0.97 F631.	N510 G43 H5 Z25.0 M8
N140 G43 H10 Z25.0 M8	N325 X26.905 Y31.01 Z-	N515 G8 P1
N145 G8 P1	1.442	N520 Z2.0
N150 Z2.0	N330 X25.0 Y36.7 Z-1.915	N525 G1 Z-0.97 F631.
N155 G1 Z-0.97 F630.	N335 X26.905 Y31.01 Z-	N530 X9.5 Z-1.723
N160 X23.045 Y18.973 Z-	2.387	N535 X19.5 Z-2.477
1.442	N340 X25.0 Y36.7 Z-2.859	N540 X9.5 Z-3.23
N165 X25.0 Y13.3 Z-1.915	N345 X26.905 Y31.01 Z-	N545 X19.5 Z-3.983
N170 X23.045 Y18.973 Z-	3.332	N550 X9.5 Z-4.737
2.387	N350 X25.0 Y36.7 Z-3.804	N555 X19.5 Z-5.49
N175 X25.0 Y13.3 Z-2.859	N355 X26.905 Y31.01 Z-	N560 X9.5 Z-6.243
N180 X23.045 Y18.973 Z-	4.276	N565 X19.5 Z-6.997
3.332	N360 X25.0 Y36.7 Z-4.749	N570 X9.5 Z-7.75
N185 X25.0 Y13.3 Z-3.804	N365 X26.905 Y31.01 Z-	N575 X40.5 F1261.
N190 X23.045 Y18.973 Z-	5.221	N580 Y27.5
4.276	N370 X25.0 Y36.7 Z-5.694	N585 X9.5
N195 X25.0 Y13.3 Z-4.749	N375 X26.905 Y31.01 Z-	N590 Y22.5
N200 X23.045 Y18.973 Z-	6.166	N595 G0 Z25.0
5.221	N380 X25.0 Y36.7 Z-6.638	N600 (RECTANGULAR
N205 X25.0 Y13.3 Z-5.694	N385 X26.905 Y31.01 Z-	POCKET FINISH
N210 X23.045 Y18.973 Z-	7.111	RECT_POCK2)
6.166	N390 X25.0 Y36.7 Z-7.583	

N605 G0 G17 G94 X9.5 Y22.5 Z25.0 S9702 N610 X19.5 N615 Z2.0 N620 G1 Z-7.72 F582. N625 X9.5 Z-8.147 N630 X19.5 Z-8.573 N635 X9.5 Z-9.0 N640 X40.5 F1164. N645 Y27.5 N650 X9.5 N655 Y22.5 N660 X10.252 Y22.128 N665 G3 X11.081 Y22.0 I0.829 J2.622 F582. N670 G1 X41.0 F1164. N675 Y28.0 N680 X9.0 N685 Y22.0 N690 X14.081 N695 G3 X14.91 Y22.128 I0. J2.75 F582. N700 G1 X15.662 Y22.5 F1164. N705 G0 Z25.0 N710 (RECTANGULAR POCKET ROUGH1 RECT_POCK1) N715 G0 G17 G94 X22.99 Y69.01 Z25.0 S6306 N720 Z2.0 N725 G1 Z-0.97 F631. N730 Y79.01 Z-1.751 N735 Y69.01 Z-2.532 N740 Y79.01 Z-3.313 N745 Y69.01 Z-4.094 N750 Y79.01 Z-4.876 N755 Y69.01 Z-5.657 N760 Y79.01 Z-6.438 N765 Y69.01 Z-7.219 N770 Y79.01 Z-8.0 N775 Y65.99 F1261. N780 X27.01 N785 Y79.01 N790 X22.99 N795 G3 X20.13 Y76.816 I0. J-2.96 N800 X19.66 Y73.242 I13.337 J-3.574 N805 G1 Y62.66 N810 X30.34 N815 Y82.34 N820 X19.66 N825 Y79.01	N830 G2 X17.995 Y76.126 I-3.33 J0. N835 G3 X16.33 Y73.242 I1.665 J-2.884 N840 G1 Y59.33 N845 X33.67 N850 Y85.67 N855 X16.33 N860 Y79.01 N865 G2 X14.665 Y76.126 I-3.33 J0. N870 G3 X13.0 Y73.242 I1.665 J-2.884 N875 G1 Y56.0 N880 X37.0 N885 Y89.0 N890 X13.0 N895 Y73.242 N900 G0 Z25.0 N905 M9 N915 M5 N920 (RECTANGULAR POCKET RECT_POCK1) N925 G40 G80 G90 G94 N930 T4 D4 M6 (ENDMILLM0800:REG 8.0) N935 M1 N940 G0 G17 X24.0 Y70.0 S9000 M3 N945 G43 H4 Z25.0 M8 N950 G8 P1 N955 Z2.0 N960 G1 Z-7.97 F432. N965 Y78.0 Z-8.313 N970 Y70.0 Z-8.657 N975 Y78.0 Z-9.0 N980 Y67.0 F864. N985 X26.0 N990 Y78.0 N995 X24.0 N1000 G3 X20.565 Y75.364 I0. J-3.556 F432. N1005 X20.0 Y71.072 I16.02 J-4.293 F696. N1010 G1 Y63.0 F864. N1015 X30.0 N1020 Y82.0 N1025 X20.0 N1030 Y78.0 N1035 G2 X18.0 Y74.536 I-4.0 J0. F1296. N1040 G3 X16.0 Y71.072 I2.0 J-3.464 F432. N1045 G1 Y59.0 F864. N1050 X34.0	N1055 Y86.0 N1060 X16.0 N1065 Y78.0 N1070 G2 X14.0 Y74.536 I-4.0 J0. F1296. N1075 G3 X12.0 Y71.072 I2.0 J-3.464 F432. N1080 G1 Y56.0 F864. N1085 G3 X13.0 Y55.0 I1.0 J0. F432. N1090 G1 X37.0 F864. N1095 G3 X38.0 Y56.0 I0. J1.0 F432. N1100 G1 Y89.0 F864. N1105 G3 X37.0 Y90.0 I- 1.0 J0. F432. N1110 G1 X13.0 F864. N1115 G3 X12.0 Y89.0 I0. J-1.0 F432. N1120 G1 Y71.072 F864. N1125 X11.266 Y70.277 N1130 G3 X11.0 Y69.228 I1.934 J-1.049 F432. N1135 G1 Y56.0 F864. N1140 G3 X13.0 Y54.0 I2.0 J0. F432. N1145 G1 X37.0 F864. N1150 G3 X39.0 Y56.0 I0. J2.0 F432. N1155 G1 Y89.0 F864. N1160 G3 X37.0 Y91.0 I- 2.0 J0. F432. N1165 G1 X13.0 F864. N1170 G3 X11.0 Y89.0 I0. J-2.0 F432. N1175 G1 Y66.228 F864. N1180 G3 X11.266 Y65.179 I2.2 J0. F432. N1185 G1 X12.0 Y64.384 F864. N1190 G0 Z25.0 N1195 M9 N1205 M5 N1210 (SIDE SIDE2) N1215 G40 G80 G90 G94 N1220 T2 D2 M6 (ENDMILLM1600:REG 16.0) N1225 M1 N1230 G0 G17 X60.056 Y107.621 S3941 M3 N1235 G43 H2 Z25.0 M8 N1240 G8 P1 N1245 Z2.0 N1250 G1 Z-6.0 F631.
--	---	---

N1255 X55.656 Y100.0
 F1261.
 N1260 Y0.
 N1265 X60.056 Y-7.621
 N1270 X54.728
 N1275 X50.328 Y0.
 N1280 Y100.0
 N1285 X54.728 Y107.621
 N1290 X49.4
 N1295 X45.0 Y100.0
 N1300 Y0.
 N1305 X49.4 Y-7.621
 N1310 G0 Z25.0
 N1315 M9
 N1325 M5
 N1330 (SLOT SLOT1)
 N1335 G40 G80 G90 G94
 N1340 T4 D4 M6
 (ENDMILLM0800:REG 8.0)
 N1345 M1
 N1350 G0 G17 X44.0
 Y72.5 S7882 M3
 N1355 G43 H4 Z25.0 M8
 N1360 G8 P1
 N1365 Z-3.0
 N1370 G1 Z-5.97 F631.
 N1375 X36.0 Z-6.529
 N1380 X44.0 Z-7.088
 N1385 X36.0 Z-7.647
 N1390 X44.0 Z-8.206
 N1395 X36.0 Z-8.764
 N1400 X44.0 Z-9.323
 N1405 X36.0 Z-9.882
 N1410 X44.0 Z-10.441
 N1415 X36.0 Z-11.0
 N1420 X51.0 F1261.
 N1425 G0 Z25.0
 N1430 M9
 N1440 M5
 N1455 M30%

C.1.3.2. ISO 6983 part programme for Siemens 840D controller – Part III

;S_Test3_RF	N215X23.045Y18.973Z-	N400X26.905Y31.01Z-
;5-2-2012	4.276	6.166
N20G54 SUPA D0	N220X25.0Y13.3Z-4.749	N405X25.0Y36.7Z-6.638
N25G17 G21 G94 G90	N225X23.045Y18.973Z-	N410X26.905Y31.01Z-
G64 SOFT	5.221	7.111
N30 ;TOOL	N230X25.0Y13.3Z-5.694	N415X25.0Y36.7Z-7.583
CHANGE(facemillM6600)	N235X23.045Y18.973Z-	N420X26.905Y31.01Z-
N35 ;FACE FINISH FACE1	6.166	8.055
N40T1	N240X25.0Y13.3Z-6.638	N425X25.0Y36.7Z-8.528
N45M6	N245X23.045Y18.973Z-	N430X26.905Y31.01Z-9.0
N50D1	7.111	N435X25.0Y36.7F1262.
N55M01	N250X25.0Y13.3Z-7.583	N440X23.545Y37.917
N60S8000M3	N255X23.045Y18.973Z-	N445G2X23.0Y39.733
N65 G4F4	8.055	I=AC(26.3) J=AC(39.733)
N70G0G54.1P1X83.0Y20.	N260X25.0Y13.3Z-8.528	N450G1Y40.0
3	N265X23.045Y18.973Z-	N455X27.0
N75Z25.0M8	9.0	N460Y31.0
N80Z3.0	N270X25.0Y13.3F1261.	N465X23.0
N85G1Z-1.0F8065.	N275X26.455Y14.517	N470Y40.0
N90X-33.0	N280G3X27.0Y16.333	N475X25.733
N95Y73.6	I=AC(23.7) J=AC(16.333)	N480G2X26.187Y39.864
N100X83.0	N285G1Y19.0	I=AC(25.733) J=AC(39.175)
N105G0Z25.0	N290X23.0	N485G1X26.491Y39.5
N110Z150M9	N295Y10.0	N490G0Z25.0
N115X0Y0M5	N300X27.0	N495Z150M9
N120 ;TOOL	N305Y19.0	N500X0Y0M5
CHANGE(endmillM1200:r	N310X26.667	N505 ;TOOL
eg)	N315G3X24.85Y18.455	CHANGE(endmillM1000:r
N125 ;RECTANGULAR	I=AC(26.667) J=AC(15.7)	eg)
POCKET ROUGH1	N320G1X23.634Y17.0	N510 ;RECTANGULAR
RECT_POCK4	N325G0Z25.0	POCKET ROUGH1
N130T2	N330G94	RECT_POCK2
N135M6	N335X25.0Y36.7S5256F63	N515T3
N140D2	1.	N520M6
N145 M01	N340Z2.0	N525D3
N150G94	N345G1Z-0.97	N530 M01
N155S5255M3	N350X26.905Y31.01Z-	N535G94
N160 G4F4	1.442	N540S6306M3
N165G0X25.0Y13.3	N355X25.0Y36.7Z-1.915	N545 G4F4
N170Z25.0M8	N360X26.905Y31.01Z-	N550G0X19.5Y22.5
N175Z2.0	2.387	N555Z25.0M8
N180G1Z-0.97F630.	N365X25.0Y36.7Z-2.859	N560Z2.0
N185X23.045Y18.973Z-	N370X26.905Y31.01Z-	N565G1Z-0.97F631.
1.442	3.332	N570X9.5Z-1.723
N190X25.0Y13.3Z-1.915	N375X25.0Y36.7Z-3.804	N575X19.5Z-2.477
N195X23.045Y18.973Z-	N380X26.905Y31.01Z-	N580X9.5Z-3.23
2.387	4.276	N585X19.5Z-3.983
N200X25.0Y13.3Z-2.859	N385X25.0Y36.7Z-4.749	N590X9.5Z-4.737
N205X23.045Y18.973Z-	N390X26.905Y31.01Z-	N595X19.5Z-5.49
3.332	5.221	N600X9.5Z-6.243
N210X25.0Y13.3Z-3.804	N395X25.0Y36.7Z-5.694	N605X19.5Z-6.997
		N610X9.5Z-7.75

N615X40.5F1261.	N865Y79.01	N1090G2X18.0Y74.536
N620Y27.5	N870G2X17.995Y76.126	I=AC(16.0) J=AC(78.0)
N625X9.5	I=AC(16.33) J=AC(79.01)	N1095G3X16.0Y71.072
N630Y22.5	N875G3X16.33Y73.242	I=AC(20.0) J=AC(71.072)
N635G0Z25.0	I=AC(19.66) J=AC(73.242)	N1100G1Y59.0
N640G94	N880G1Y59.33	N1105X34.0
N645S9702F582.	N885X33.67	N1110Y86.0
N650X19.5	N890Y85.67	N1115X16.0
N655Z2.0	N895X16.33	N1120Y78.0
N660G1Z-7.72	N900Y79.01	N1125G2X14.0Y74.536
N665X9.5Z-8.147	N905G2X14.665Y76.126	I=AC(12.0) J=AC(78.0)
N670X19.5Z-8.573	I=AC(13.0) J=AC(79.01)	N1130G3X12.0Y71.072
N675X9.5Z-9.0	N910G3X13.0Y73.242	I=AC(16.0) J=AC(71.072)
N680X40.5F1164.	I=AC(16.33) J=AC(73.242)	N1135G1Y56.0
N685Y27.5	N915G1Y56.0	N1140G3X13.0Y55.0
N690X9.5	N920X37.0	I=AC(13.0) J=AC(56.0)
N695Y22.5	N925Y89.0	N1145G1X37.0
N700X10.252Y22.128	N930X13.0	N1150G3X38.0Y56.0
N705G3X11.081Y22.0	N935Y73.242	I=AC(37.0) J=AC(56.0)
I=AC(11.081) J=AC(24.75)	N940G0Z25.0	N1155G1Y89.0
N710G1X41.0	N945Z150M9	N1160G3X37.0Y90.0
N715Y28.0	N950X0Y0M5	I=AC(37.0) J=AC(89.0)
N720X9.0	N955 ;TOOL	N1165G1X13.0
N725Y22.0	CHANGE(endmillM0800:r	N1170G3X12.0Y89.0
N730X14.081	eg)	I=AC(13.0) J=AC(89.0)
N735G3X14.91Y22.128	N960 ;RECTANGULAR	N1175G1Y71.072
I=AC(14.081) J=AC(24.75)	POCKET FINISH	N1180X11.266Y70.277
N740G1X15.662Y22.5	RECT_POCK1	N1185G3X11.0Y69.228
N745G0Z25.0	N965T4	I=AC(13.2) J=AC(69.228)
N750G94	N970M6	N1190G1Y56.0
N755X22.99Y69.01S6306F	N975D4	N1195G3X13.0Y54.0
631.	N980 M01	I=AC(13.0) J=AC(56.0)
N760Z2.0	N985G94	N1200G1X37.0
N765G1Z-0.97	N990S9000M3	N1205G3X39.0Y56.0
N770Y79.01Z-1.751	N995 G4F4	I=AC(37.0) J=AC(56.0)
N775Y69.01Z-2.532	N1000G0X24.0Y70.0	N1210G1Y89.0
N780Y79.01Z-3.313	N1005Z25.0M8	N1215G3X37.0Y91.0
N785Y69.01Z-4.094	N1010Z2.0	I=AC(37.0) J=AC(89.0)
N790Y79.01Z-4.876	N1015G1Z-7.97F432.	N1220G1X13.0
N795Y69.01Z-5.657	N1020Y78.0Z-8.313	N1225G3X11.0Y89.0
N800Y79.01Z-6.438	N1025Y70.0Z-8.657	I=AC(13.0) J=AC(89.0)
N805Y69.01Z-7.219	N1030Y78.0Z-9.0	N1230G1Y66.228
N810Y79.01Z-8.0	N1035Y67.0F864.	N1235G3X11.266Y65.179
N815Y65.99F1261.	N1040X26.0	I=AC(13.2) J=AC(66.228)
N820X27.01	N1045Y78.0	N1240G1X12.0Y64.384
N825Y79.01	N1050X24.0	N1245G0Z25.0
N830X22.99	N1055G3X20.565Y75.364	N1250Z150M9
N835G3X20.13Y76.816	I=AC(24.0) J=AC(74.444)	N1255X0Y0M5
I=AC(22.99) J=AC(76.05)	N1060X20.0Y71.072	N1260 ;TOOL
N840X19.66Y73.242	I=AC(36.585) J=AC(71.072)	CHANGE(endmillM1600:r
I=AC(33.467) J=AC(73.242)	N1065G1Y63.0	eg)
N845G1Y62.66	N1070X30.0	N1265 ;SIDE ROUGH1
N850X30.34	N1075Y82.0	SIDE2
N855Y82.34	N1080X20.0	N1270T5
N860X19.66	N1085Y78.0	N1275M6

N1280D5	N1530M30
N1285 M01	N1535%
N1290G94	
N1295S3941M3	
N1300 G4F4	
N1305G0X60.056Y107.62	
1	
N1310Z25.0M8	
N1315Z2.0	
N1320G1Z-6.0F631.	
N1325X55.656Y100.0F126	
1.	
N1330Y0.	
N1335X60.056Y-7.621	
N1340X54.728	
N1345X50.328Y0.	
N1350Y100.0	
N1355X54.728Y107.621	
N1360X49.4	
N1365X45.0Y100.0	
N1370Y0.	
N1375X49.4Y-7.621	
N1380G0Z25.0	
N1385Z150M9	
N1390X0Y0M5	
N1395 ;TOOL	
CHANGE(endmillM0800:r	
eg)	
N1400 ;SLOT ROUGH1	
SLOT1	
N1405T4	
N1410M6	
N1415D4	
N1420 M01	
N1425G94	
N1430S7882M3	
N1435 G4F4	
N1440G0X44.0Y72.5	
N1445Z25.0M8	
N1450Z-3.0	
N1455G1Z-5.97F631.	
N1460X36.0Z-6.529	
N1465X44.0Z-7.088	
N1470X36.0Z-7.647	
N1475X44.0Z-8.206	
N1480X36.0Z-8.764	
N1485X44.0Z-9.323	
N1490X36.0Z-9.882	
N1495X44.0Z-10.441	
N1500X36.0Z-11.0	
N1505X51.0F1261.	
N1510G0Z25.0	
N1515Z150M9	
N1520X0Y0M5	
N1525G54 SUPA D0	

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('GENERATED ISO 14649-11 FILE','AUTOMATIC OUTPUT OF UPCI FROM A CNC
PART PROGRAMME'), '1');
FILE_NAME('EXAMPLE.STP', '2012-04-17', ('XIANZHI ZHANG'),('UNIVERSITY OF BATH,
BATH,UK'),$, '$', 'ISO 14649',$);
FILE_SCHEMA(('MACHINING_SCHEMA','MILLING_SCHEMA'));
ENDSEC;

DATA;
#1=PROJECT('RECOGNISED ISO 14649 PART 21 FILE FROM G&M CODES',#2,(#3),$$,$$,$);
#2=WORKPLAN('MAIN WORKPLAN',( #4,#5,#6,#7,#8),$,$9,$);
#3=WORKPIECE('WORKPIECE50.0X100.0X20.0',$,$,$,$,$21,());
#4=MACHINING_WORKINGSTEP('WS PLANAR_FACE1',#10,#11,#12,$);
#5=MACHINING_WORKINGSTEP('WS DRILLING ROUND HOLE 6.34',#10,#44,#45,$);
#6=MACHINING_WORKINGSTEP('WS REAMING ROUND HOLE 6.34',#10,#44,#46,$);
#7=MACHINING_WORKINGSTEP('WS ROUGH POCKET2',#10,#69,#70,$);
#8=MACHINING_WORKINGSTEP('WS FINISH POCKET2',#10,#69,#71,$);
#9=SETUP('SETUP',#150,#10,(#151));
#10=ELEMENTARY_SURFACE('SECURITY PLANE',#13);
#11=PLANAR_FACE('PLANAR_FACE1',#3,(#12),#17,#18,#19,#20,$,());
#12=PLANE_ROUGH_MILLING($,$,'PLANAR_FACE1',$,$,$26,$27,$28,$,$29,$30,$31,$,$);
#13=AXIS2_PLACEMENT_3D('SECURITY PLANE PLACEMENT',#14,#15,#16);
#14=CARTESIAN_POINT('SECURITY PLANE: LOCATION',(0.0,0.0,10.0,0.0,0.0));
#15=DIRECTION('AXIS',(0.0,0.0,1.0));
#16=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#17=AXIS2_PLACEMENT_3D('PLANAR_FACE1 PLACEMENT',#34,#35,#36);
#18=ELEMENTARY_SURFACE('PLANAR_FACE1 DEPTH PLANE',#37);
#19=LINEAR_PATH($,$41,#42);
#20=LINEAR_PROFILE($,$43);
#21=BLOCK('WORKPIECE BLOCK',#22,50.0,100.0,-20.0);
#22=AXIS2_PLACEMENT_3D('WORKPIECE BLOCK PLACEMENT',#23,#24,#25);
#23=CARTESIAN_POINT('WORKPIECE BLOCK: LOCATION',(0.0,0.0,0.0));
#24=DIRECTION('AXIS',(0.0,0.0,1.0));
#25=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#26=MILLING_CUTTING_TOOL('T8',#32,(),$,$,$);
#27=MILLING_TECHNOLOGY(0.016666666666666666,.TCP.,$,133.33333333333334,$,$,$,$);
#28=MILLING_MACHINE_FUNCTIONS(.F.,$,$,$,$,(),$,$,$,());
#29=PLUNGE_RAMP($,$);
#30=PLUNGE_RAMP($,$);
#31=BIDIRECTIONAL($,$,$,$,$);
#32=FACEMILL(#33,$,$,$,$);
#33=MILLING_TOOL_DIMENSION(66.0,$,$,$,0.0,$,$);
#34=CARTESIAN_POINT('PLANAR_FACE1',(90.0,20.0,0.0));
#35=DIRECTION('AXIS',(0.0,0.0,1.0));
#36=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#37=AXIS2_PLACEMENT_3D('PLANAR_FACE1 DEPTH',#38,#39,#40);
#38=CARTESIAN_POINT('PLANAR_FACE1 DEPTH',(0.0,0.30000000000000007,-1.0));
#39=DIRECTION('AXIS',(0.0,0.0,1.0));
```

227

```

#92=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#93=AXIS2_PLACEMENT_3D('POCKET2 DEPTH',#94,#95,#96);
#94=CARTESIAN_POINT('POCKET2 DEPTH',(12.5,0.5,-8.030000000000001));
#95=DIRECTION('AXIS',(0.0,0.0,1.0));
#96=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#97=COMPOSITE_CURVE('BOUNDARY: POCKET2',(#98,#99,#100,#101,#102,#103,#104,#105),.F.);
#98=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#106);
#99=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#109);
#100=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#117);
#101=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#120);
#102=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#128);
#103=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#131);
#104=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#139);
#105=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#142);
#106=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#107,#108));
#107=CARTESIAN_POINT('POLYLINE POINT 1',(13.0,34.0,7.97));
#108=CARTESIAN_POINT('POLYLINE POINT 1',(0.0,34.0,7.97));
#109=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#110,(#111),(#112),.T.,.CARTESIAN.);
#110=CIRCLE('CIRCLE',#113,6.0);
#111=CARTESIAN_POINT('TRIM POINT 1',(0.0,34.0,7.97));
#112=CARTESIAN_POINT('TRIM POINT 2',(-6.0,28.0,7.97));
#113=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#114,#115,#116);
#114=CARTESIAN_POINT('CIRCLE CENTER',(0.0,28.0,7.97));
#115=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#116=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#117=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#118,#119));
#118=CARTESIAN_POINT('POLYLINE POINT 1',(-6.0,28.0,7.97));
#119=CARTESIAN_POINT('POLYLINE POINT 1',(-6.0,0.0,7.97));
#120=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#121,(#122),(#123),.T.,.CARTESIAN.);
#121=CIRCLE('CIRCLE',#124,6.0);
#122=CARTESIAN_POINT('TRIM POINT 1',(-6.0,0.0,7.97));
#123=CARTESIAN_POINT('TRIM POINT 2',(0.0,-6.0,7.97));
#124=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#125,#126,#127);
#125=CARTESIAN_POINT('CIRCLE CENTER',(0.0,0.0,7.97));
#126=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#127=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#128=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#129,#130));
#129=CARTESIAN_POINT('POLYLINE POINT 1',(0.0,-6.0,7.97));
#130=CARTESIAN_POINT('POLYLINE POINT 1',(13.0,-6.0,7.97));
#131=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#132,(#133),(#134),.T.,.CARTESIAN.);
#132=CIRCLE('CIRCLE',#135,6.0);
#133=CARTESIAN_POINT('TRIM POINT 1',(13.0,-6.0,7.97));
#134=CARTESIAN_POINT('TRIM POINT 2',(19.0,0.0,7.97));
#135=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#136,#137,#138);
#136=CARTESIAN_POINT('CIRCLE CENTER',(13.0,0.0,7.97));
#137=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#138=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#139=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#140,#141));
#140=CARTESIAN_POINT('POLYLINE POINT 1',(19.0,0.0,7.97));
#141=CARTESIAN_POINT('POLYLINE POINT 1',(19.0,28.0,7.97));
#142=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#143,(#144),(#145),.T.,.CARTESIAN.);

```



```
#143=CIRCLE('CIRCLE',#146,6.0);
#144=CARTESIAN_POINT('TRIM POINT 1',(19.0,28.0,7.97));
#145=CARTESIAN_POINT('TRIM POINT 2',(13.0,34.0,7.97));
#146=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#147,#148,#149);
#147=CARTESIAN_POINT('CIRCLE CENTER',(13.0,28.0,7.97));
#148=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#149=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#150=AXIS2_PLACEMENT_3D('SETUP ORIGIN',#152,#153,#154);
#151=WORKPIECE_SETUP(#3,#155,$,$,());
#152=CARTESIAN_POINT('SETUP LOCATION',(0.0,0.0,0.0));
#153=DIRECTION('AXIS',(0.0,0.0,1.0));
#154=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#155=AXIS2_PLACEMENT_3D('WORKPIECE50.0X100.0X20.0 SETUP',#156,#157,#158);
#156=CARTESIAN_POINT('WORKPIECE50.0X100.0X20.0SETUP LOCATION',(0.0,0.0,0.0));
#157=DIRECTION('AXIS',(0.0,0.0,1.0));
#158=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
ENDSEC;
END-ISO-10303-21;
```

230

```

#43=NUMERIC_PARAMETER('null',119.3,'mm');
#44=ROUND_HOLE('ROUND HOLE 6.34',#3,(#45,#46),#47,#48,#49,$,$);
#45=MULTISTEP_DRILLING($,$,'DRILLING ROUND HOLE
6.34',,$,$50,#51,#52,$,$,$,$,$53,0.0,6.3,6.3,$);
#46=REAMING($,$,'REAMING ROUND HOLE 6.34',,$,$56,#57,#58,$,$,$,$,$59,.F.,,$,$);
#47=AXIS2_PLACEMENT_3D('ROUND HOLE 6.34 PLACEMENT',#62,#63,#64);
#48=ELEMENTARY_SURFACE('ROUND HOLE 6.34 DEPTH PLANE',#65);
#49=TOLERANCED_LENGTH_MEASURE(6.34,$);
#50=MILLING_CUTTING_TOOL('T2',#54,(),$,$,$);
#51=MILLING_TECHNOLOGY(0.0058333333333333334,.TCP.,$,50.0,$,$,$,$);
#52=MILLING_MACHINE_FUNCTIONS(.F.,,$,$,$,$(),$,$,$,$());
#53=DRILLING_TYPE_STRATEGY($,$,$,$,$);
#54=TWIST_DRILL(#55,$,$,$,$);
#55=MILLING_TOOL_DIMENSION(6.2,$,$,$0.0,$,$);
#56=MILLING_CUTTING_TOOL('T3',#60,(),$,$,$);
#57=MILLING_TECHNOLOGY(0.0182,.TCP.,$,31.883333333333333,$,$,$,$,$);
#58=MILLING_MACHINE_FUNCTIONS(.F.,,$,$,$,$(),$,$,$,$());
#59=DRILLING_TYPE_STRATEGY($,$,$,$,$);
#60=REAMER(#61,$,$,$,$);
#61=MILLING_TOOL_DIMENSION(6.34,$,$,$0.0,$,$);
#62=CARTESIAN_POINT('ROUND HOLE 6.34 LOCATION',(10.0,50.0,0.0));
#63=DIRECTION('AXIS',(0.0,0.0,1.0));
#64=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#65=AXIS2_PLACEMENT_3D('ROUND HOLE 6.34 DEPTH',#66,#67,#68);
#66=CARTESIAN_POINT('ROUND HOLE 6.34 DEPTH',(0.0,0.0,-11.0));
#67=DIRECTION('AXIS',(0.0,0.0,1.0));
#68=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#69=CLOSED_POCKET('POCKET2',#3,(#70,#71),#72,#73,(),$,$74,#75,$,$76);
#70=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'ROUGH
POCKET2',,$,$77,#78,#79,$,$80,#81,#82,$,$0.5,1.0);
#71=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'FINISH
POCKET2',,$,$77,#85,#86,$,$87,#88,#89,$,$0.0,0.0);
#72=AXIS2_PLACEMENT_3D('POCKET2 PLACEMENT',#90,#91,#92);
#73=ELEMENTARY_SURFACE('POCKET2 DEPTH PLANE',#93);
#74=PLANAR_POCKET_BOTTOM_CONDITION();
#75=TOLERANCED_LENGTH_MEASURE(0.0,$);
#76=GENERAL_CLOSED_PROFILE($,$97);
#77=MILLING_CUTTING_TOOL('T4',#83,(),$,$,$);
#78=MILLING_TECHNOLOGY(0.0133333333333333334,.TCP.,$,66.66666666666667,$,$,$,$,$);
#79=MILLING_MACHINE_FUNCTIONS(.F.,,$,$,$,$(),$,$,$,$());
#80=PLUNGE_ZIGZAG($,$,$);
#81=PLUNGE_TOOLAXIS($);
#82=BIDIRECTIONAL_CONTOUR($,$,$,$,$,$);
#83=ENDMILL(#84,$,$,$,$);
#84=MILLING_TOOL_DIMENSION(12.0,$,$,$0.0,$,$);
#85=MILLING_TECHNOLOGY(0.0083333333333333333,.TCP.,$,83.33333333333333,$,$,$,$,$);
#86=MILLING_MACHINE_FUNCTIONS(.F.,,$,$,$,$(),$,$,$,$());
#87=PLUNGE_ZIGZAG($,$,$);
#88=PLUNGE_TOOLAXIS($);
#89=BIDIRECTIONAL_CONTOUR($,$,$,$,$,$);
#90=CARTESIAN_POINT('POCKET2',(26.0,36.0,-0.97));
#91=DIRECTION('AXIS',(0.0,0.0,1.0));
#92=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#93=AXIS2_PLACEMENT_3D('POCKET2 DEPTH',#94,#95,#96);
#94=CARTESIAN_POINT('POCKET2 DEPTH',(12.5,0.5,-8.030000000000001));

```

```

#95=DIRECTION('AXIS',(0.0,0.0,1.0));
#96=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#97=COMPOSITE_CURVE('BOUNDARY: POCKET2',(98,99,100,101,102,103,104,105),.F.);
#98=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#106);
#99=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#109);
#100=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#117);
#101=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#120);
#102=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#128);
#103=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#131);
#104=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#139);
#105=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#142);
#106=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#107,#108));
#107=CARTESIAN_POINT('POLYLINE POINT 1',(13.0,34.0,7.97));
#108=CARTESIAN_POINT('POLYLINE POINT 1',(0.0,34.0,7.97));
#109=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#110,(#111),(#112),.T.,.CARTESIAN.);
#110=CIRCLE('CIRCLE',#113,6.0);
#111=CARTESIAN_POINT('TRIM POINT 1',(0.0,34.0,7.97));
#112=CARTESIAN_POINT('TRIM POINT 2',(-6.0,28.0,7.97));
#113=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#114,#115,#116);
#114=CARTESIAN_POINT('CIRCLE CENTER',(0.0,28.0,7.97));
#115=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#116=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#117=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#118,#119));
#118=CARTESIAN_POINT('POLYLINE POINT 1',(-6.0,28.0,7.97));
#119=CARTESIAN_POINT('POLYLINE POINT 1',(-6.0,0.0,7.97));
#120=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#121,(#122),(#123),.T.,.CARTESIAN.);
#121=CIRCLE('CIRCLE',#124,6.0);
#122=CARTESIAN_POINT('TRIM POINT 1',(-6.0,0.0,7.97));
#123=CARTESIAN_POINT('TRIM POINT 2',(0.0,-6.0,7.97));
#124=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#125,#126,#127);
#125=CARTESIAN_POINT('CIRCLE CENTER',(0.0,0.0,7.97));
#126=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#127=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#128=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#129,#130));
#129=CARTESIAN_POINT('POLYLINE POINT 1',(0.0,-6.0,7.97));
#130=CARTESIAN_POINT('POLYLINE POINT 1',(13.0,-6.0,7.97));
#131=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#132,(#133),(#134),.T.,.CARTESIAN.);
#132=CIRCLE('CIRCLE',#135,6.0);
#133=CARTESIAN_POINT('TRIM POINT 1',(13.0,-6.0,7.97));
#134=CARTESIAN_POINT('TRIM POINT 2',(19.0,0.0,7.97));
#135=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#136,#137,#138);
#136=CARTESIAN_POINT('CIRCLE CENTER',(13.0,0.0,7.97));
#137=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#138=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#139=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#140,#141));
#140=CARTESIAN_POINT('POLYLINE POINT 1',(19.0,0.0,7.97));
#141=CARTESIAN_POINT('POLYLINE POINT 1',(19.0,28.0,7.97));
#142=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#143,(#144),(#145),.T.,.CARTESIAN.);
#143=CIRCLE('CIRCLE',#146,6.0);
#144=CARTESIAN_POINT('TRIM POINT 1',(19.0,28.0,7.97));
#145=CARTESIAN_POINT('TRIM POINT 2',(13.0,34.0,7.97));

```

```
#146=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#147,#148,#149);
#147=CARTESIAN_POINT('CIRCLE CENTER',(13.0,28.0,7.97));
#148=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#149=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#150=AXIS2_PLACEMENT_3D('SETUP ORIGIN',#152,#153,#154);
#151=WORKPIECE_SETUP(#3,#155,$,$,());
#152=CARTESIAN_POINT('SETUP LOCATION',(0.0,0.0,0.0));
#153=DIRECTION('AXIS',(0.0,0.0,1.0));
#154=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#155=AXIS2_PLACEMENT_3D('WORKPIECE50.0X100.0X20.0 SETUP',#156,#157,#158);
#156=CARTESIAN_POINT('WORKPIECE50.0X100.0X20.0SETUP LOCATION',(0.0,0.0,0.0));
#157=DIRECTION('AXIS',(0.0,0.0,1.0));
#158=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
ENDSEC;

END-ISO-10303-21;
```

234

235

```

#93=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#116);
#94=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#124);
#95=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#127);
#96=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#135);
#97=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#138);
#98=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#146);
#99=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#149);
#100=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#157);
#101=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#160);
#102=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#103,#104));
#103=CARTESIAN_POINT('POLYLINE POINT 1',(40.0,99.0,0.0));
#104=CARTESIAN_POINT('POLYLINE POINT 2',(40.0,44.0,0.0));
#105=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#106,(#107),(#108),.T.,.CARTESIAN.);
#106=CIRCLE('CIRCLE',#109,5.0);
#107=CARTESIAN_POINT('TRIM POINT 1',(40.0,44.0,0.0));
#108=CARTESIAN_POINT('TRIM POINT 2',(35.0,39.0,0.0));
#109=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#110,#111,#112);
#110=CARTESIAN_POINT('CIRCLE CENTER',(35.0,44.0,0.0));
#111=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#112=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#113=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#114,#115));
#114=CARTESIAN_POINT('POLYLINE POINT 1',(35.0,39.0,0.0));
#115=CARTESIAN_POINT('POLYLINE POINT 2',(20.0,39.0,0.0));
#116=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#117,(#118),(#119),.T.,.CARTESIAN.);
#117=CIRCLE('CIRCLE',#120,5.0);
#118=CARTESIAN_POINT('TRIM POINT 1',(20.0,39.0,0.0));
#119=CARTESIAN_POINT('TRIM POINT 2',(15.0,44.0,0.0));
#120=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#121,#122,#123);
#121=CARTESIAN_POINT('CIRCLE CENTER',(20.0,44.0,0.0));
#122=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#123=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#124=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#125,#126));
#125=CARTESIAN_POINT('POLYLINE POINT 1',(15.0,44.0,0.0));
#126=CARTESIAN_POINT('POLYLINE POINT 2',(15.0,69.0,0.0));
#127=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#128,(#129),(#130),.T.,.CARTESIAN.);
#128=CIRCLE('CIRCLE',#131,5.0);
#129=CARTESIAN_POINT('TRIM POINT 1',(15.0,69.0,0.0));
#130=CARTESIAN_POINT('TRIM POINT 2',(10.0,74.0,0.0));
#131=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#132,#133,#134);
#132=CARTESIAN_POINT('CIRCLE CENTER',(10.0,69.0,0.0));
#133=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#134=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#135=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#136,#137));
#136=CARTESIAN_POINT('POLYLINE POINT 1',(10.0,74.0,0.0));
#137=CARTESIAN_POINT('POLYLINE POINT 2',(5.0,74.0,0.0));
#138=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#139,(#140),(#141),.T.,.CARTESIAN.);
#139=CIRCLE('CIRCLE',#142,5.0);
#140=CARTESIAN_POINT('TRIM POINT 1',(5.0,74.0,0.0));
#141=CARTESIAN_POINT('TRIM POINT 2',(0.0,79.0,0.0));
#142=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#143,#144,#145);
#143=CARTESIAN_POINT('CIRCLE CENTER',(5.0,79.0,0.0));

```



```

#144=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#145=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#146=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#147,#148));
#147=CARTESIAN_POINT('POLYLINE POINT 1',(0.0,79.0,0.0));
#148=CARTESIAN_POINT('POLYLINE POINT 2',(0.0,99.0,0.0));
#149=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#150,(#151),(#152),.T.,.CARTESIAN.);
#150=CIRCLE('CIRCLE',#153,5.0);
#151=CARTESIAN_POINT('TRIM POINT 1',(0.0,99.0,0.0));
#152=CARTESIAN_POINT('TRIM POINT 2',(5.0,104.0,0.0));
#153=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#154,#155,#156);
#154=CARTESIAN_POINT('CIRCLE CENTER',(5.0,99.0,0.0));
#155=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#156=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#157=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#158,#159));
#158=CARTESIAN_POINT('POLYLINE POINT 1',(5.0,104.0,0.0));
#159=CARTESIAN_POINT('POLYLINE POINT 2',(35.0,104.0,0.0));
#160=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#161,(#162),(#163),.T.,.CARTESIAN.);
#161=CIRCLE('CIRCLE',#164,5.0);
#162=CARTESIAN_POINT('TRIM POINT 1',(35.0,104.0,0.0));
#163=CARTESIAN_POINT('TRIM POINT 2',(40.0,99.0,0.0));
#164=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#165,#166,#167);
#165=CARTESIAN_POINT('CIRCLE CENTER',(35.0,99.0,0.0));
#166=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#167=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#168=CLOSED_POCKET('POCKET2',#3,(#169,#170),#171,#172,(),$, #173,#174,$,#175);
#169=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'ROUGH
POCKET2',$,$,#176,#177,#178,$,#179,#180,#181,$,$,1.0,1.0);
#170=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'FINISH
POCKET2',$,$,#176,#184,#185,$,#186,#187,#188,$,$,0.0,0.0);
#171=AXIS2_PLACEMENT_3D('POCKET2 PLACEMENT',#189,#190,#191);
#172=ELEMENTARY_SURFACE('POCKET2 DEPTH PLANE',#192);
#173=PLANAR_POCKET_BOTTOM_CONDITION();
#174=TOLERANCED_LENGTH_MEASURE(0.0,$);
#175=GENERAL_CLOSED_PROFILE($,#196);
#176=MILLING_CUTTING_TOOL('T4',#182,(),$,$,$);
#177=MILLING_TECHNOLOGY(0.0210166666666666666666666666667,$,131.36666666666666666666666666667,$,$,$,$,$);
#178=MILLING_MACHINE_FUNCTIONS(.F.,$,$,$,$,(),$,$,$,());
#179=PLUNGE_ZIGZAG($,$,$);
#180=PLUNGE_TOOLAXIS($);
#181=CONTOUR_SPIRAL($,$,$,$);
#182=ENDMILL(#183,$,$,$,$);
#183=MILLING_TOOL_DIMENSION(8.0,$,$,$,0.0,$,$);
#184=MILLING_TECHNOLOGY(0.0144,.TCP.,$,150.0,$,$,$,$,$);
#185=MILLING_MACHINE_FUNCTIONS(.F.,$,$,$,$,(),$,$,$,());
#186=PLUNGE_ZIGZAG($,$,$);
#187=PLUNGE_TOOLAXIS($);
#188=CONTOUR_SPIRAL($,$,$,$);
#189=CARTESIAN_POINT('POCKET2',(23.0,81.0,-0.97));
#190=DIRECTION('AXIS',(0.0,0.0,1.0));
#191=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#192=AXIS2_PLACEMENT_3D('POCKET2 DEPTH',#193,#194,#195);
#193=CARTESIAN_POINT('POCKET2 DEPTH',(8.0,0.0,-8.0300000000000001));
#194=DIRECTION('AXIS',(0.0,0.0,1.0));

```

```

#195=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#196=COMPOSITE_CURVE('BOUNDARY:
POCKET2',(197,198,199,200,201,202,203,204,205,206,207,208),.F.);
#197=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#209);
#198=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#217);
#199=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#220);
#200=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#228);
#201=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#231);
#202=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#239);
#203=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#242);
#204=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#250);
#205=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#253);
#206=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#261);
#207=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#264);
#208=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#272);
#209=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#210,(#211),(#212),.T.,.CARTESIAN.);
#210=CIRCLE('CIRCLE',#213,5.0);
#211=CARTESIAN_POINT('TRIM POINT 1',(-8.0,9.0,7.97));
#212=CARTESIAN_POINT('TRIM POINT 2',(-13.0,4.0,7.97));
#213=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#214,#215,#216);
#214=CARTESIAN_POINT('CIRCLE CENTER',(-8.0,4.0,7.97));
#215=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#216=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#217=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#218,#219));
#218=CARTESIAN_POINT('POLYLINE POINT 1',(-13.0,4.0,7.97));
#219=CARTESIAN_POINT('POLYLINE POINT 1',(-13.0,-6.0,7.97));
#220=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#221,(#222),(#223),.T.,.CARTESIAN.);
#221=CIRCLE('CIRCLE',#224,5.0);
#222=CARTESIAN_POINT('TRIM POINT 1',(-13.0,-6.0,7.97));
#223=CARTESIAN_POINT('TRIM POINT 2',(-8.0,-11.0,7.97));
#224=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#225,#226,#227);
#225=CARTESIAN_POINT('CIRCLE CENTER',(-8.0,-6.0,7.97));
#226=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#227=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#228=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#229,#230));
#229=CARTESIAN_POINT('POLYLINE POINT 1',(-8.0,-11.0,7.97));
#230=CARTESIAN_POINT('POLYLINE POINT 1',(-3.0,-11.0,7.97));
#231=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#232,(#233),(#234),.T.,.CARTESIAN.);
#232=CIRCLE('CIRCLE',#235,5.0);
#233=CARTESIAN_POINT('TRIM POINT 1',(-3.0,-11.0,7.97));
#234=CARTESIAN_POINT('TRIM POINT 2',(2.0,-16.0,7.97));
#235=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#236,#237,#238);
#236=CARTESIAN_POINT('CIRCLE CENTER',(-3.0,-16.0,7.97));
#237=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#238=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#239=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#240,#241));
#240=CARTESIAN_POINT('POLYLINE POINT 1',(2.0,-16.0,7.97));
#241=CARTESIAN_POINT('POLYLINE POINT 1',(2.0,-41.0,7.97));
#242=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#243,(#244),(#245),.T.,.CARTESIAN.);
#243=CIRCLE('CIRCLE',#246,5.0);
#244=CARTESIAN_POINT('TRIM POINT 1',(2.0,-41.0,7.97));

```

```

#245=CARTESIAN_POINT('TRIM POINT 2',(7.0,-46.0,7.97));
#246=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#247,#248,#249);
#247=CARTESIAN_POINT('CIRCLE CENTER',(7.0,-41.0,7.97));
#248=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#249=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#250=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#251,#252));
#251=CARTESIAN_POINT('POLYLINE POINT 1',(7.0,-46.0,7.97));
#252=CARTESIAN_POINT('POLYLINE POINT 1',(12.0,-46.0,7.97));
#253=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#254,(#255),(#256),.T.,.CARTESIAN.);
#254=CIRCLE('CIRCLE',#257,5.0);
#255=CARTESIAN_POINT('TRIM POINT 1',(12.0,-46.0,7.97));
#256=CARTESIAN_POINT('TRIM POINT 2',(17.0,-41.0,7.97));
#257=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#258,#259,#260);
#258=CARTESIAN_POINT('CIRCLE CENTER',(12.0,-41.0,7.97));
#259=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#260=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#261=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#262,#263));
#262=CARTESIAN_POINT('POLYLINE POINT 1',(17.0,-41.0,7.97));
#263=CARTESIAN_POINT('POLYLINE POINT 1',(17.0,4.0,7.97));
#264=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#265,(#266),(#267),.T.,.CARTESIAN.);
#265=CIRCLE('CIRCLE',#268,5.0);
#266=CARTESIAN_POINT('TRIM POINT 1',(17.0,4.0,7.97));
#267=CARTESIAN_POINT('TRIM POINT 2',(12.0,9.0,7.97));
#268=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#269,#270,#271);
#269=CARTESIAN_POINT('CIRCLE CENTER',(12.0,4.0,7.97));
#270=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#271=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#272=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#273,#274));
#273=CARTESIAN_POINT('POLYLINE POINT 1',(12.0,9.0,7.97));
#274=CARTESIAN_POINT('POLYLINE POINT 1',(-8.0,9.0,7.97));
#275=CLOSED_POCKET('POCKET3',#3,(#276),#277,#278,(),$, #279,#280,$, #281);
#276=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'POCKET3',$,$, #176,#282,#283,$, #284,#285,#286,$
,$,$,$);
#277=AXIS2_PLACEMENT_3D('POCKET3 PLACEMENT',#287,#288,#289);
#278=ELEMENTARY_SURFACE('POCKET3 DEPTH PLANE',#290);
#279=PLANAR_POCKET_BOTTOM_CONDITION();
#280=TOLERANCED_LENGTH_MEASURE(0.0,$);
#281=CIRCULAR_CLOSED_PROFILE(#294,#295);
#282=MILLING_TECHNOLOGY(0.021033333333333334,.TCP.,$,131.41666666666666,$,$,$,$,$);
#283=MILLING_MACHINE_FUNCTIONS(.F.,$, $,$,$,(),$, $,$,());
#284=PLUNGE_HELIX($,$,$);
#285=PLUNGE_TOOLAXIS($);
#286=CONTOUR_SPIRAL($,$,$,$);
#287=CARTESIAN_POINT('POCKET3',(16.0,16.0,-4.97));
#288=DIRECTION('AXIS',(0.0,0.0,1.0));
#289=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#290=AXIS2_PLACEMENT_3D('POCKET3 DEPTH',#291,#292,#293);
#291=CARTESIAN_POINT('POCKET3 DEPTH',(4.0,0.0,-5.03));
#292=DIRECTION('AXIS',(0.0,0.0,1.0));
#293=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#294=AXIS2_PLACEMENT_3D('CIRCULAR PROFILE LOCATION',#296,#297,#298);
#295=TOLERANCED_LENGTH_MEASURE(24.0,$);
#296=CARTESIAN_POINT('LOCATION POINT',(0.0,0.0,0.0));

```

240

241

243

244


```

#93=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#116);
#94=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#124);
#95=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#127);
#96=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#135);
#97=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#138);
#98=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#146);
#99=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#149);
#100=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#157);
#101=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,T.,#160);
#102=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#103,#104));
#103=CARTESIAN_POINT('POLYLINE POINT 1',(40.0,99.0,0.0));
#104=CARTESIAN_POINT('POLYLINE POINT 2',(40.0,44.0,0.0));
#105=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#106,(#107),(#108),.T.,.CARTESIAN.);
#106=CIRCLE('CIRCLE',#109,5.0);
#107=CARTESIAN_POINT('TRIM POINT 1',(40.0,44.0,0.0));
#108=CARTESIAN_POINT('TRIM POINT 2',(35.0,39.0,0.0));
#109=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#110,#111,#112);
#110=CARTESIAN_POINT('CIRCLE CENTER',(35.0,44.0,0.0));
#111=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#112=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#113=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#114,#115));
#114=CARTESIAN_POINT('POLYLINE POINT 1',(35.0,39.0,0.0));
#115=CARTESIAN_POINT('POLYLINE POINT 2',(20.0,39.0,0.0));
#116=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#117,(#118),(#119),.T.,.CARTESIAN.);
#117=CIRCLE('CIRCLE',#120,5.0);
#118=CARTESIAN_POINT('TRIM POINT 1',(20.0,39.0,0.0));
#119=CARTESIAN_POINT('TRIM POINT 2',(15.0,44.0,0.0));
#120=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#121,#122,#123);
#121=CARTESIAN_POINT('CIRCLE CENTER',(20.0,44.0,0.0));
#122=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#123=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#124=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#125,#126));
#125=CARTESIAN_POINT('POLYLINE POINT 1',(15.0,44.0,0.0));
#126=CARTESIAN_POINT('POLYLINE POINT 2',(15.0,69.0,0.0));
#127=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#128,(#129),(#130),.T.,.CARTESIAN.);
#128=CIRCLE('CIRCLE',#131,5.0);
#129=CARTESIAN_POINT('TRIM POINT 1',(15.0,69.0,0.0));
#130=CARTESIAN_POINT('TRIM POINT 2',(10.0,74.0,0.0));
#131=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#132,#133,#134);
#132=CARTESIAN_POINT('CIRCLE CENTER',(10.0,69.0,0.0));
#133=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#134=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#135=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#136,#137));
#136=CARTESIAN_POINT('POLYLINE POINT 1',(10.0,74.0,0.0));
#137=CARTESIAN_POINT('POLYLINE POINT 2',(5.0,74.0,0.0));
#138=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#139,(#140),(#141),.T.,.CARTESIAN.);
#139=CIRCLE('CIRCLE',#142,5.0);
#140=CARTESIAN_POINT('TRIM POINT 1',(5.0,74.0,0.0));
#141=CARTESIAN_POINT('TRIM POINT 2',(0.0,79.0,0.0));
#142=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#143,#144,#145);
#143=CARTESIAN_POINT('CIRCLE CENTER',(5.0,79.0,0.0));

```

```

#144=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#145=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#146=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#147,#148));
#147=CARTESIAN_POINT('POLYLINE POINT 1',(0.0,79.0,0.0));
#148=CARTESIAN_POINT('POLYLINE POINT 2',(0.0,99.0,0.0));
#149=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#150,(#151),(#152),.T.,.CARTESIAN.);
#150=CIRCLE('CIRCLE',#153,5.0);
#151=CARTESIAN_POINT('TRIM POINT 1',(0.0,99.0,0.0));
#152=CARTESIAN_POINT('TRIM POINT 2',(5.0,104.0,0.0));
#153=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#154,#155,#156);
#154=CARTESIAN_POINT('CIRCLE CENTER',(5.0,99.0,0.0));
#155=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#156=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#157=POLYLINE('POLYLINE FOR CONTOUR: PLANAR_FACE3',(#158,#159));
#158=CARTESIAN_POINT('POLYLINE POINT 1',(5.0,104.0,0.0));
#159=CARTESIAN_POINT('POLYLINE POINT 2',(35.0,104.0,0.0));
#160=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
PLANAR_FACE3',#161,(#162),(#163),.T.,.CARTESIAN.);
#161=CIRCLE('CIRCLE',#164,5.0);
#162=CARTESIAN_POINT('TRIM POINT 1',(35.0,104.0,0.0));
#163=CARTESIAN_POINT('TRIM POINT 2',(40.0,99.0,0.0));
#164=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#165,#166,#167);
#165=CARTESIAN_POINT('CIRCLE CENTER',(35.0,99.0,0.0));
#166=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#167=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#168=CLOSED_POCKET('POCKET2',#3,(#169,#170),#171,#172,(),$, #173,#174,$,#175);
#169=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'ROUGH
POCKET2',$,$,#176,#177,#178,$,#179,#180,#181,$,$,1.0,1.0);
#170=BOTTOM_AND_SIDE_FINISH_MILLING($,$,'FINISH
POCKET2',$,$,#176,#184,#185,$,#186,#187,#188,$,$,0.0,0.0);
#171=AXIS2_PLACEMENT_3D('POCKET2 PLACEMENT',#189,#190,#191);
#172=ELEMENTARY_SURFACE('POCKET2 DEPTH PLANE',#192);
#173=PLANAR_POCKET_BOTTOM_CONDITION();
#174=TOLERANCED_LENGTH_MEASURE(0.0,$);
#175=GENERAL_CLOSED_PROFILE($,#196);
#176=MILLING_CUTTING_TOOL('T3',#182,(),$,$,$);
#177=MILLING_TECHNOLOGY(0.021016666666666666666666666667,$,131.36666666666666666666666666667,$,$,$,$,$);
#178=MILLING_MACHINE_FUNCTIONS(.F.,$,$,$,$,(),$,$,$,());
#179=PLUNGE_ZIGZAG($,$,$);
#180=PLUNGE_TOOLAXIS($);
#181=CONTOUR_SPIRAL($,$,$,$);
#182=ENDMILL(#183,$,$,$,$);
#183=MILLING_TOOL_DIMENSION(8.0,$,$,$,0.0,$,$);
#184=MILLING_TECHNOLOGY(0.0144,.TCP.,$,150.0,$,$,$,$,$);
#185=MILLING_MACHINE_FUNCTIONS(.F.,$,$,$,$,(),$,$,$,());
#186=PLUNGE_ZIGZAG($,$,$);
#187=PLUNGE_TOOLAXIS($);
#188=CONTOUR_SPIRAL($,$,$,$);
#189=CARTESIAN_POINT('POCKET2',(23.0,81.0,-0.97));
#190=DIRECTION('AXIS',(0.0,0.0,1.0));
#191=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#192=AXIS2_PLACEMENT_3D('POCKET2 DEPTH',#193,#194,#195);
#193=CARTESIAN_POINT('POCKET2 DEPTH',(8.0,0.0,-8.0300000000000001));
#194=DIRECTION('AXIS',(0.0,0.0,1.0));

```

```

#195=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#196=COMPOSITE_CURVE('BOUNDARY:
POCKET2',(197,198,199,200,201,202,203,204,205,206,207,208),.F.);
#197=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#209);
#198=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#217);
#199=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#220);
#200=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#228);
#201=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#231);
#202=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#239);
#203=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#242);
#204=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#250);
#205=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#253);
#206=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#261);
#207=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#264);
#208=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#272);
#209=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#210,211,212),.T.,.CARTESIAN.);
#210=CIRCLE('CIRCLE',#213,5.0);
#211=CARTESIAN_POINT('TRIM POINT 1',(-8.0,9.0,7.97));
#212=CARTESIAN_POINT('TRIM POINT 2',(-13.0,4.0,7.97));
#213=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#214,215,216);
#214=CARTESIAN_POINT('CIRCLE CENTER',(-8.0,4.0,7.97));
#215=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#216=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#217=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',218,219));
#218=CARTESIAN_POINT('POLYLINE POINT 1',(-13.0,4.0,7.97));
#219=CARTESIAN_POINT('POLYLINE POINT 1',(-13.0,-6.0,7.97));
#220=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#221,222,223),.T.,.CARTESIAN.);
#221=CIRCLE('CIRCLE',#224,5.0);
#222=CARTESIAN_POINT('TRIM POINT 1',(-13.0,-6.0,7.97));
#223=CARTESIAN_POINT('TRIM POINT 2',(-8.0,-11.0,7.97));
#224=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#225,226,227);
#225=CARTESIAN_POINT('CIRCLE CENTER',(-8.0,-6.0,7.97));
#226=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#227=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#228=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',229,230));
#229=CARTESIAN_POINT('POLYLINE POINT 1',(-8.0,-11.0,7.97));
#230=CARTESIAN_POINT('POLYLINE POINT 1',(-3.0,-11.0,7.97));
#231=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#232,233,234),.T.,.CARTESIAN.);
#232=CIRCLE('CIRCLE',#235,5.0);
#233=CARTESIAN_POINT('TRIM POINT 1',(-3.0,-11.0,7.97));
#234=CARTESIAN_POINT('TRIM POINT 2',(2.0,-16.0,7.97));
#235=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#236,237,238);
#236=CARTESIAN_POINT('CIRCLE CENTER',(-3.0,-16.0,7.97));
#237=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#238=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#239=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',240,241));
#240=CARTESIAN_POINT('POLYLINE POINT 1',(2.0,-16.0,7.97));
#241=CARTESIAN_POINT('POLYLINE POINT 1',(2.0,-41.0,7.97));
#242=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#243,244,245),.T.,.CARTESIAN.);
#243=CIRCLE('CIRCLE',#246,5.0);
#244=CARTESIAN_POINT('TRIM POINT 1',(2.0,-41.0,7.97));

```

```

#245=CARTESIAN_POINT('TRIM POINT 2',(7.0,-46.0,7.97));
#246=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#247,#248,#249);
#247=CARTESIAN_POINT('CIRCLE CENTER',(7.0,-41.0,7.97));
#248=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#249=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#250=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#251,#252));
#251=CARTESIAN_POINT('POLYLINE POINT 1',(7.0,-46.0,7.97));
#252=CARTESIAN_POINT('POLYLINE POINT 1',(12.0,-46.0,7.97));
#253=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#254,(#255),(#256),.T.,.CARTESIAN.);
#254=CIRCLE('CIRCLE',#257,5.0);
#255=CARTESIAN_POINT('TRIM POINT 1',(12.0,-46.0,7.97));
#256=CARTESIAN_POINT('TRIM POINT 2',(17.0,-41.0,7.97));
#257=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#258,#259,#260);
#258=CARTESIAN_POINT('CIRCLE CENTER',(12.0,-41.0,7.97));
#259=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#260=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#261=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#262,#263));
#262=CARTESIAN_POINT('POLYLINE POINT 1',(17.0,-41.0,7.97));
#263=CARTESIAN_POINT('POLYLINE POINT 1',(17.0,4.0,7.97));
#264=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#265,(#266),(#267),.T.,.CARTESIAN.);
#265=CIRCLE('CIRCLE',#268,5.0);
#266=CARTESIAN_POINT('TRIM POINT 1',(17.0,4.0,7.97));
#267=CARTESIAN_POINT('TRIM POINT 2',(12.0,9.0,7.97));
#268=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#269,#270,#271);
#269=CARTESIAN_POINT('CIRCLE CENTER',(12.0,4.0,7.97));
#270=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#271=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#272=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#273,#274));
#273=CARTESIAN_POINT('POLYLINE POINT 1',(12.0,9.0,7.97));
#274=CARTESIAN_POINT('POLYLINE POINT 1',(-8.0,9.0,7.97));
#275=CLOSED_POCKET('POCKET3',#3,(#276),#277,#278,(),$, #279,#280,$, #281);
#276=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'POCKET3',$,$, #176,#282,#283,$, #284,#285,#286,$
,$,$,$);
#277=AXIS2_PLACEMENT_3D('POCKET3 PLACEMENT',#287,#288,#289);
#278=ELEMENTARY_SURFACE('POCKET3 DEPTH PLANE',#290);
#279=PLANAR_POCKET_BOTTOM_CONDITION();
#280=TOLERANCED_LENGTH_MEASURE(0.0,$);
#281=CIRCULAR_CLOSED_PROFILE(#294,#295);
#282=MILLING_TECHNOLOGY(0.021033333333333334,.TCP.,$, #131.41666666666666,$,$,$,$,$);
#283=MILLING_MACHINE_FUNCTIONS(.F.,$, $,$,$,(),$, $,$,());
#284=PLUNGE_HELIX($,$,$);
#285=PLUNGE_TOOLAXIS($);
#286=CONTOUR_SPIRAL($,$,$,$);
#287=CARTESIAN_POINT('POCKET3',(16.0,16.0,-4.97));
#288=DIRECTION('AXIS',(0.0,0.0,1.0));
#289=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#290=AXIS2_PLACEMENT_3D('POCKET3 DEPTH',#291,#292,#293);
#291=CARTESIAN_POINT('POCKET3 DEPTH',(4.0,0.0,-5.03));
#292=DIRECTION('AXIS',(0.0,0.0,1.0));
#293=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#294=AXIS2_PLACEMENT_3D('CIRCULAR PROFILE LOCATION',#296,#297,#298);
#295=TOLERANCED_LENGTH_MEASURE(24.0,$);
#296=CARTESIAN_POINT('LOCATION POINT',(0.0,0.0,0.0));

```

249

```
#350=ROUND_HOLE('ROUND HOLE 3.4'#3,(#351),#352,#353,#354,$,$);
#351=MULTISTEP_DRILLING($,'DRILLING ROUND HOLE
3.4',$,$,#322,#355,#356,$,$,$,$,#357,0.0,3.0,3.0,$);
#352=AXIS2_PLACEMENT_3D('ROUND HOLE 3.4 PLACEMENT',#358,#359,#360);
#353=ELEMENTARY_SURFACE('ROUND HOLE 3.4 DEPTH PLANE',#361);
#354=TOLERANCED_LENGTH_MEASURE(3.4,$);
#355=MILLING_TECHNOLOGY(0.00606666666666666666..TCP.,$,134.75,$,$,$,$,$);
#356=MILLING_MACHINE_FUNCTIONS(.F.,$,$,$,$,$,$,$,$,$,$());
#357=DRILLING_TYPE_STRATEGY($,$,$,$,$,$,$);
#358=CARTESIAN_POINT('ROUND HOLE 3.4 LOCATION',(9.0,69.0,0.0));
#359=DIRECTION('AXIS',(0.0,0.0,1.0));
#360=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#361=AXIS2_PLACEMENT_3D('ROUND HOLE 3.4 DEPTH',#362,#363,#364);
#362=CARTESIAN_POINT('ROUND HOLE 3.4 DEPTH',(0.0,0.0,-10.901));
#363=DIRECTION('AXIS',(0.0,0.0,1.0));
#364=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#365=ROUND_HOLE('ROUND HOLE 3.4'#3,(#366),#367,#368,#369,$,$);
#366=MULTISTEP_DRILLING($,'DRILLING ROUND HOLE
3.4',$,$,#322,#370,#371,$,$,$,$,$,#372,0.0,3.0,3.0,$);
#367=AXIS2_PLACEMENT_3D('ROUND HOLE 3.4 PLACEMENT',#373,#374,#375);
#368=ELEMENTARY_SURFACE('ROUND HOLE 3.4 DEPTH PLANE',#376);
#369=TOLERANCED_LENGTH_MEASURE(3.4,$);
#370=MILLING_TECHNOLOGY(0.00606666666666666666..TCP.,$,134.75,$,$,$,$,$,$);
#371=MILLING_MACHINE_FUNCTIONS(.F.,$,$,$,$,$,$,$,$,$,$());
#372=DRILLING_TYPE_STRATEGY($,$,$,$,$,$,$);
#373=CARTESIAN_POINT('ROUND HOLE 3.4 LOCATION',(21.0,66.0,0.0));
#374=DIRECTION('AXIS',(0.0,0.0,1.0));
#375=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#376=AXIS2_PLACEMENT_3D('ROUND HOLE 3.4 DEPTH',#377,#378,#379);
#377=CARTESIAN_POINT('ROUND HOLE 3.4 DEPTH',(0.0,0.0,-10.901));
#378=DIRECTION('AXIS',(0.0,0.0,1.0));
#379=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#380=ROUND_HOLE('ROUND HOLE 3.4'#3,(#381),#382,#383,#384,$,$);
#381=MULTISTEP_DRILLING($,'DRILLING ROUND HOLE
3.4',$,$,#322,#385,#386,$,$,$,$,$,#387,0.0,3.0,3.0,$);
#382=AXIS2_PLACEMENT_3D('ROUND HOLE 3.4 PLACEMENT',#388,#389,#390);
#383=ELEMENTARY_SURFACE('ROUND HOLE 3.4 DEPTH PLANE',#391);
#384=TOLERANCED_LENGTH_MEASURE(3.4,$);
#385=MILLING_TECHNOLOGY(0.00606666666666666666..TCP.,$,134.75,$,$,$,$,$,$);
#386=MILLING_MACHINE_FUNCTIONS(.F.,$,$,$,$,$,$,$,$,$,$());
#387=DRILLING_TYPE_STRATEGY($,$,$,$,$,$,$);
#388=CARTESIAN_POINT('ROUND HOLE 3.4 LOCATION',(24.0,34.0,0.0));
#389=DIRECTION('AXIS',(0.0,0.0,1.0));
#390=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#391=AXIS2_PLACEMENT_3D('ROUND HOLE 3.4 DEPTH',#392,#393,#394);
#392=CARTESIAN_POINT('ROUND HOLE 3.4 DEPTH',(0.0,0.0,-10.901));
#393=DIRECTION('AXIS',(0.0,0.0,1.0));
#394=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#395=ROUND_HOLE('ROUND HOLE 3.4'#3,(#396),#397,#398,#399,$,$);
#396=MULTISTEP_DRILLING($,'DRILLING ROUND HOLE
3.4',$,$,#322,#400,#401,$,$,$,$,$,$,#402,0.0,3.0,3.0,$);
#397=AXIS2_PLACEMENT_3D('ROUND HOLE 3.4 PLACEMENT',#403,#404,#405);
#398=ELEMENTARY_SURFACE('ROUND HOLE 3.4 DEPTH PLANE',#406);
#399=TOLERANCED_LENGTH_MEASURE(3.4,$);
#400=MILLING_TECHNOLOGY(0.00606666666666666666..TCP.,$,134.75,$,$,$,$,$,$);
```


252

253

```

#96=CARTESIAN_POINT('TRIM POINT 2',(-2.0,-9.0,0.97));
#97=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#98,#99,#100);
#98=CARTESIAN_POINT('CIRCLE CENTER',(-2.0,-3.0,0.97));
#99=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#100=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#101=POLYLINE('POLYLINE FOR CONTOUR: POCKET1',(#102,#103));
#102=CARTESIAN_POINT('POLYLINE POINT 1',(-2.0,-9.0,0.97));
#103=CARTESIAN_POINT('POLYLINE POINT 1',(2.0,-9.0,0.97));
#104=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET1',#105,(#106),(#107),.T.,.CARTESIAN.);
#105=CIRCLE('CIRCLE',#108,6.0);
#106=CARTESIAN_POINT('TRIM POINT 1',(2.0,-9.0,0.97));
#107=CARTESIAN_POINT('TRIM POINT 2',(8.0,-3.0,0.97));
#108=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#109,#110,#111);
#109=CARTESIAN_POINT('CIRCLE CENTER',(2.0,-3.0,0.97));
#110=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#111=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#112=POLYLINE('POLYLINE FOR CONTOUR: POCKET1',(#113,#114));
#113=CARTESIAN_POINT('POLYLINE POINT 1',(8.0,-3.0,0.97));
#114=CARTESIAN_POINT('POLYLINE POINT 1',(8.0,6.0,0.97));
#115=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET1',#116,(#117),(#118),.T.,.CARTESIAN.);
#116=CIRCLE('CIRCLE',#119,6.0);
#117=CARTESIAN_POINT('TRIM POINT 1',(8.0,6.0,0.97));
#118=CARTESIAN_POINT('TRIM POINT 2',(2.0,12.0,0.97));
#119=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#120,#121,#122);
#120=CARTESIAN_POINT('CIRCLE CENTER',(2.0,6.0,0.97));
#121=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#122=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#123=CLOSED_POCKET('POCKET2',#3,(#124),#125,#126,(),$, #127,#128,$, #129);
#124=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'POCKET2',$,$, #55,#130,#131,$, #132,#133,#134,$,
$, $, $);
#125=AXIS2_PLACEMENT_3D('POCKET2 PLACEMENT',#135,#136,#137);
#126=ELEMENTARY_SURFACE('POCKET2 DEPTH PLANE',#138);
#127=PLANAR_POCKET_BOTTOM_CONDITION();
#128=TOLERANCED_LENGTH_MEASURE(0.0,$);
#129=GENERAL_CLOSED_PROFILE($, #142);
#130=MILLING_TECHNOLOGY(0.021033333333333334,.TCP.,$, 87.6,$,$,$,$);
#131=MILLING_MACHINE_FUNCTIONS(.F.,$, $,$,$,(), $,$,$,());
#132=PLUNGE_ZIGZAG($,$,$);
#133=PLUNGE_TOOLAXIS($);
#134=CONTOUR_SPIRAL($,$,$,$);
#135=CARTESIAN_POINT('POCKET2',(25.0,36.0,-0.97));
#136=DIRECTION('AXIS',(0.0,0.0,1.0));
#137=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#138=AXIS2_PLACEMENT_3D('POCKET2 DEPTH',#139,#140,#141);
#139=CARTESIAN_POINT('POCKET2 DEPTH',(1.90500000000000011,-4.9899999999999998,-8.03));
#140=DIRECTION('AXIS',(0.0,0.0,1.0));
#141=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#142=COMPOSITE_CURVE('BOUNDARY: POCKET2',(#143,#144,#145,#146,#147,#148,#149,#150),.F.);
#143=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#151);
#144=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#154);
#145=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#162);
#146=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#165);
#147=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#173);

```

```

#148=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,,T.,#176);
#149=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,,T.,#184);
#150=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,,T.,#187);
#151=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#152,#153));
#152=CARTESIAN_POINT('POLYLINE POINT 1',(2.0,10.0,0.97));
#153=CARTESIAN_POINT('POLYLINE POINT 1',(-2.0,10.0,0.97));
#154=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#155,(#156),(#157),.T.,.CARTESIAN.);
#155=CIRCLE('CIRCLE',#158,6.0);
#156=CARTESIAN_POINT('TRIM POINT 1',(-2.0,10.0,0.97));
#157=CARTESIAN_POINT('TRIM POINT 2',(-8.0,4.0,0.97));
#158=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#159,#160,#161);
#159=CARTESIAN_POINT('CIRCLE CENTER',(-2.0,4.0,0.97));
#160=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#161=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#162=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#163,#164));
#163=CARTESIAN_POINT('POLYLINE POINT 1',(-8.0,4.0,0.97));
#164=CARTESIAN_POINT('POLYLINE POINT 1',(-8.0,-5.0,0.97));
#165=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#166,(#167),(#168),.T.,.CARTESIAN.);
#166=CIRCLE('CIRCLE',#169,6.0);
#167=CARTESIAN_POINT('TRIM POINT 1',(-8.0,-5.0,0.97));
#168=CARTESIAN_POINT('TRIM POINT 2',(-2.0,-11.0,0.97));
#169=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#170,#171,#172);
#170=CARTESIAN_POINT('CIRCLE CENTER',(-2.0,-5.0,0.97));
#171=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#172=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#173=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#174,#175));
#174=CARTESIAN_POINT('POLYLINE POINT 1',(-2.0,-11.0,0.97));
#175=CARTESIAN_POINT('POLYLINE POINT 1',(2.0,-11.0,0.97));
#176=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#177,(#178),(#179),.T.,.CARTESIAN.);
#177=CIRCLE('CIRCLE',#180,6.0);
#178=CARTESIAN_POINT('TRIM POINT 1',(2.0,-11.0,0.97));
#179=CARTESIAN_POINT('TRIM POINT 2',(8.0,-5.0,0.97));
#180=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#181,#182,#183);
#181=CARTESIAN_POINT('CIRCLE CENTER',(2.0,-5.0,0.97));
#182=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#183=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#184=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#185,#186));
#185=CARTESIAN_POINT('POLYLINE POINT 1',(8.0,-5.0,0.97));
#186=CARTESIAN_POINT('POLYLINE POINT 1',(8.0,4.0,0.97));
#187=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#188,(#189),(#190),.T.,.CARTESIAN.);
#188=CIRCLE('CIRCLE',#191,6.0);
#189=CARTESIAN_POINT('TRIM POINT 1',(8.0,4.0,0.97));
#190=CARTESIAN_POINT('TRIM POINT 2',(2.0,10.0,0.97));
#191=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#192,#193,#194);
#192=CARTESIAN_POINT('CIRCLE CENTER',(2.0,4.0,0.97));
#193=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#194=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#195=CLOSED_POCKET('POCKET4',#3,(#196,#197),#198,#199,(),$,$,#200,#201,$,#202);
#196=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'ROUGH
POCKET4',$,$,#203,#204,#205,$,#206,#207,#208,$,$,0.5,1.25);

```

256

257

```

#300=AXIS2_PLACEMENT_3D('POCKET6 DEPTH',#301,#302,#303);
#301=CARTESIAN_POINT('POCKET6 DEPTH',(0.0,8.0,-8.030000000000001));
#302=DIRECTION('AXIS',(0.0,0.0,1.0));
#303=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#304=COMPOSITE_CURVE('BOUNDARY: POCKET6',(#305,#306,#307,#308,#309,#310,#311,#312),.F.);
#305=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#313);
#306=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#316);
#307=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#324);
#308=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#327);
#309=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#335);
#310=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#338);
#311=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#346);
#312=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#349);
#313=POLYLINE('POLYLINE FOR CONTOUR: POCKET6',(#314,#315));
#314=CARTESIAN_POINT('POLYLINE POINT 1',(13.0,25.0,7.97));
#315=CARTESIAN_POINT('POLYLINE POINT 1',(-11.0,25.0,7.97));
#316=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET6',#317,(#318),(#319),.T.,.CARTESIAN.);
#317=CIRCLE('CIRCLE',#320,6.0);
#318=CARTESIAN_POINT('TRIM POINT 1',(-11.0,25.0,7.97));
#319=CARTESIAN_POINT('TRIM POINT 2',(-17.0,19.0,7.97));
#320=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#321,#322,#323);
#321=CARTESIAN_POINT('CIRCLE CENTER',(-11.0,19.0,7.97));
#322=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#323=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#324=POLYLINE('POLYLINE FOR CONTOUR: POCKET6',(#325,#326));
#325=CARTESIAN_POINT('POLYLINE POINT 1',(-17.0,19.0,7.97));
#326=CARTESIAN_POINT('POLYLINE POINT 1',(-17.0,-14.0,7.97));
#327=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET6',#328,(#329),(#330),.T.,.CARTESIAN.);
#328=CIRCLE('CIRCLE',#331,6.0);
#329=CARTESIAN_POINT('TRIM POINT 1',(-17.0,-14.0,7.97));
#330=CARTESIAN_POINT('TRIM POINT 2',(-11.0,-20.0,7.97));
#331=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#332,#333,#334);
#332=CARTESIAN_POINT('CIRCLE CENTER',(-11.0,-14.0,7.97));
#333=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#334=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#335=POLYLINE('POLYLINE FOR CONTOUR: POCKET6',(#336,#337));
#336=CARTESIAN_POINT('POLYLINE POINT 1',(-11.0,-20.0,7.97));
#337=CARTESIAN_POINT('POLYLINE POINT 1',(13.0,-20.0,7.97));
#338=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET6',#339,(#340),(#341),.T.,.CARTESIAN.);
#339=CIRCLE('CIRCLE',#342,6.0);
#340=CARTESIAN_POINT('TRIM POINT 1',(13.0,-20.0,7.97));
#341=CARTESIAN_POINT('TRIM POINT 2',(19.0,-14.0,7.97));
#342=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#343,#344,#345);
#343=CARTESIAN_POINT('CIRCLE CENTER',(13.0,-14.0,7.97));
#344=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#345=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#346=POLYLINE('POLYLINE FOR CONTOUR: POCKET6',(#347,#348));
#347=CARTESIAN_POINT('POLYLINE POINT 1',(19.0,-14.0,7.97));
#348=CARTESIAN_POINT('POLYLINE POINT 1',(19.0,19.0,7.97));
#349=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET6',#350,(#351),(#352),.T.,.CARTESIAN.);
#350=CIRCLE('CIRCLE',#353,6.0);

```

259

```
#404=CARTESIAN_POINT('SLOT1 DEPTH',(-8.0,0.5,-5.03));
#405=DIRECTION('AXIS',(0.0,0.0,1.0));
#406=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#407=AXIS2_PLACEMENT_3D('PLACEMENT OF TRAVEL COURSE: SLOT1',#410,#411,#412);
#408=TOLERANCED_LENGTH_MEASURE(15.0,$);
#409=DIRECTION('DIRECTION: SLOT1',(-1.0,0.0,0.0));
#410=CARTESIAN_POINT('POINTSLOT1',(7.0,0.5,-5.03));
#411=DIRECTION('AXIS',(0.0,0.0,1.0));
#412=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#413=AXIS2_PLACEMENT_3D('PLACEMENT OF SWEPT SHAPE: SLOT1',#416,#417,#418);
#414=TOLERANCED_LENGTH_MEASURE(8.0,$);
#415=TOLERANCED_LENGTH_MEASURE(0.0,$);
#416=CARTESIAN_POINT('POINTSLOT1',(51.0,72.5,-11.0));
#417=DIRECTION('AXIS',(0.0,0.0,1.0));
#418=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#419=AXIS2_PLACEMENT_3D('SETUP ORIGIN',#421,#422,#423);
#420=WORKPIECE_SETUP(#3,#424,$,$,());
#421=CARTESIAN_POINT('SETUP LOCATION',(0.0,0.0,0.0));
#422=DIRECTION('AXIS',(0.0,0.0,1.0));
#423=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#424=AXIS2_PLACEMENT_3D('WORKPIECE50.0X100.0X20.0 SETUP',#425,#426,#427);
#425=CARTESIAN_POINT('WORKPIECE50.0X100.0X20.0SETUP LOCATION',(0.0,0.0,0.0));
#426=DIRECTION('AXIS',(0.0,0.0,1.0));
#427=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
ENDSEC;

END-ISO-10303-21;
```

262

```

#96=CARTESIAN_POINT('TRIM POINT 2',(-2.0,-9.0,0.97));
#97=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#98,#99,#100);
#98=CARTESIAN_POINT('CIRCLE CENTER',(-2.0,-3.0,0.97));
#99=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#100=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#101=POLYLINE('POLYLINE FOR CONTOUR: POCKET1',(#102,#103));
#102=CARTESIAN_POINT('POLYLINE POINT 1',(-2.0,-9.0,0.97));
#103=CARTESIAN_POINT('POLYLINE POINT 1',(2.0,-9.0,0.97));
#104=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET1',#105,(#106),(#107),.T.,.CARTESIAN.);
#105=CIRCLE('CIRCLE',#108,6.0);
#106=CARTESIAN_POINT('TRIM POINT 1',(2.0,-9.0,0.97));
#107=CARTESIAN_POINT('TRIM POINT 2',(8.0,-3.0,0.97));
#108=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#109,#110,#111);
#109=CARTESIAN_POINT('CIRCLE CENTER',(2.0,-3.0,0.97));
#110=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#111=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#112=POLYLINE('POLYLINE FOR CONTOUR: POCKET1',(#113,#114));
#113=CARTESIAN_POINT('POLYLINE POINT 1',(8.0,-3.0,0.97));
#114=CARTESIAN_POINT('POLYLINE POINT 1',(8.0,6.0,0.97));
#115=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET1',#116,(#117),(#118),.T.,.CARTESIAN.);
#116=CIRCLE('CIRCLE',#119,6.0);
#117=CARTESIAN_POINT('TRIM POINT 1',(8.0,6.0,0.97));
#118=CARTESIAN_POINT('TRIM POINT 2',(2.0,12.0,0.97));
#119=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#120,#121,#122);
#120=CARTESIAN_POINT('CIRCLE CENTER',(2.0,6.0,0.97));
#121=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#122=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#123=CLOSED_POCKET('POCKET2',#3,(#124),#125,#126,(),$, #127,#128,$, #129);
#124=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'POCKET2',$,$, #55,#130,#131,$, #132,#133,#134,$,
$, $, $);
#125=AXIS2_PLACEMENT_3D('POCKET2 PLACEMENT',#135,#136,#137);
#126=ELEMENTARY_SURFACE('POCKET2 DEPTH PLANE',#138);
#127=PLANAR_POCKET_BOTTOM_CONDITION();
#128=TOLERANCED_LENGTH_MEASURE(0.0,$);
#129=GENERAL_CLOSED_PROFILE($, #142);
#130=MILLING_TECHNOLOGY(0.021033333333333334,.TCP.,$, #87.6,$,$,$,$);
#131=MILLING_MACHINE_FUNCTIONS(.F.,$, $,$,$,(), $,$,$,());
#132=PLUNGE_ZIGZAG($,$,$);
#133=PLUNGE_TOOLAXIS($);
#134=CONTOUR_SPIRAL($,$,$,$);
#135=CARTESIAN_POINT('POCKET2',(25.0,36.0,-0.97));
#136=DIRECTION('AXIS',(0.0,0.0,1.0));
#137=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#138=AXIS2_PLACEMENT_3D('POCKET2 DEPTH',#139,#140,#141);
#139=CARTESIAN_POINT('POCKET2 DEPTH',(1.90500000000000011,-4.9899999999999998,-8.03));
#140=DIRECTION('AXIS',(0.0,0.0,1.0));
#141=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#142=COMPOSITE_CURVE('BOUNDARY: POCKET2',(#143,#144,#145,#146,#147,#148,#149,#150),.F.);
#143=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#151);
#144=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#154);
#145=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#162);
#146=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#165);
#147=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,.T.,#173);

```

```

#148=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,,T.,#176);
#149=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,,T.,#184);
#150=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS,,T.,#187);
#151=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#152,#153));
#152=CARTESIAN_POINT('POLYLINE POINT 1',(2.0,10.0,0.97));
#153=CARTESIAN_POINT('POLYLINE POINT 1',(-2.0,10.0,0.97));
#154=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#155,(#156),(#157),.T.,.CARTESIAN.);
#155=CIRCLE('CIRCLE',#158,6.0);
#156=CARTESIAN_POINT('TRIM POINT 1',(-2.0,10.0,0.97));
#157=CARTESIAN_POINT('TRIM POINT 2',(-8.0,4.0,0.97));
#158=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#159,#160,#161);
#159=CARTESIAN_POINT('CIRCLE CENTER',(-2.0,4.0,0.97));
#160=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#161=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#162=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#163,#164));
#163=CARTESIAN_POINT('POLYLINE POINT 1',(-8.0,4.0,0.97));
#164=CARTESIAN_POINT('POLYLINE POINT 1',(-8.0,-5.0,0.97));
#165=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#166,(#167),(#168),.T.,.CARTESIAN.);
#166=CIRCLE('CIRCLE',#169,6.0);
#167=CARTESIAN_POINT('TRIM POINT 1',(-8.0,-5.0,0.97));
#168=CARTESIAN_POINT('TRIM POINT 2',(-2.0,-11.0,0.97));
#169=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#170,#171,#172);
#170=CARTESIAN_POINT('CIRCLE CENTER',(-2.0,-5.0,0.97));
#171=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#172=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#173=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#174,#175));
#174=CARTESIAN_POINT('POLYLINE POINT 1',(-2.0,-11.0,0.97));
#175=CARTESIAN_POINT('POLYLINE POINT 1',(2.0,-11.0,0.97));
#176=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#177,(#178),(#179),.T.,.CARTESIAN.);
#177=CIRCLE('CIRCLE',#180,6.0);
#178=CARTESIAN_POINT('TRIM POINT 1',(2.0,-11.0,0.97));
#179=CARTESIAN_POINT('TRIM POINT 2',(8.0,-5.0,0.97));
#180=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#181,#182,#183);
#181=CARTESIAN_POINT('CIRCLE CENTER',(2.0,-5.0,0.97));
#182=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#183=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#184=POLYLINE('POLYLINE FOR CONTOUR: POCKET2',(#185,#186));
#185=CARTESIAN_POINT('POLYLINE POINT 1',(8.0,-5.0,0.97));
#186=CARTESIAN_POINT('POLYLINE POINT 1',(8.0,4.0,0.97));
#187=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET2',#188,(#189),(#190),.T.,.CARTESIAN.);
#188=CIRCLE('CIRCLE',#191,6.0);
#189=CARTESIAN_POINT('TRIM POINT 1',(8.0,4.0,0.97));
#190=CARTESIAN_POINT('TRIM POINT 2',(2.0,10.0,0.97));
#191=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#192,#193,#194);
#192=CARTESIAN_POINT('CIRCLE CENTER',(2.0,4.0,0.97));
#193=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#194=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#195=CLOSED_POCKET('POCKET4',#3,(#196,#197),#198,#199,(),$,$,#200,#201,$,#202);
#196=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'ROUGH
POCKET4',$,$,#203,#204,#205,$,#206,#207,#208,$,$,0.5,1.25);

```

265

266

```

#300=AXIS2_PLACEMENT_3D('POCKET6 DEPTH',#301,#302,#303);
#301=CARTESIAN_POINT('POCKET6 DEPTH',(0.0,8.0,-8.030000000000001));
#302=DIRECTION('AXIS',(0.0,0.0,1.0));
#303=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#304=COMPOSITE_CURVE('BOUNDARY: POCKET6',(#305,#306,#307,#308,#309,#310,#311,#312),.F.);
#305=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#313);
#306=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#316);
#307=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#324);
#308=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#327);
#309=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#335);
#310=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#338);
#311=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#346);
#312=COMPOSITE_CURVE_SEGMENT(.CONTINUOUS.,.T.,#349);
#313=POLYLINE('POLYLINE FOR CONTOUR: POCKET6',(#314,#315));
#314=CARTESIAN_POINT('POLYLINE POINT 1',(13.0,25.0,7.97));
#315=CARTESIAN_POINT('POLYLINE POINT 1',(-11.0,25.0,7.97));
#316=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET6',#317,(#318),(#319),.T.,.CARTESIAN.);
#317=CIRCLE('CIRCLE',#320,6.0);
#318=CARTESIAN_POINT('TRIM POINT 1',(-11.0,25.0,7.97));
#319=CARTESIAN_POINT('TRIM POINT 2',(-17.0,19.0,7.97));
#320=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#321,#322,#323);
#321=CARTESIAN_POINT('CIRCLE CENTER',(-11.0,19.0,7.97));
#322=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#323=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#324=POLYLINE('POLYLINE FOR CONTOUR: POCKET6',(#325,#326));
#325=CARTESIAN_POINT('POLYLINE POINT 1',(-17.0,19.0,7.97));
#326=CARTESIAN_POINT('POLYLINE POINT 1',(-17.0,-14.0,7.97));
#327=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET6',#328,(#329),(#330),.T.,.CARTESIAN.);
#328=CIRCLE('CIRCLE',#331,6.0);
#329=CARTESIAN_POINT('TRIM POINT 1',(-17.0,-14.0,7.97));
#330=CARTESIAN_POINT('TRIM POINT 2',(-11.0,-20.0,7.97));
#331=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#332,#333,#334);
#332=CARTESIAN_POINT('CIRCLE CENTER',(-11.0,-14.0,7.97));
#333=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#334=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#335=POLYLINE('POLYLINE FOR CONTOUR: POCKET6',(#336,#337));
#336=CARTESIAN_POINT('POLYLINE POINT 1',(-11.0,-20.0,7.97));
#337=CARTESIAN_POINT('POLYLINE POINT 1',(13.0,-20.0,7.97));
#338=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET6',#339,(#340),(#341),.T.,.CARTESIAN.);
#339=CIRCLE('CIRCLE',#342,6.0);
#340=CARTESIAN_POINT('TRIM POINT 1',(13.0,-20.0,7.97));
#341=CARTESIAN_POINT('TRIM POINT 2',(19.0,-14.0,7.97));
#342=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#343,#344,#345);
#343=CARTESIAN_POINT('CIRCLE CENTER',(13.0,-14.0,7.97));
#344=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#345=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#346=POLYLINE('POLYLINE FOR CONTOUR: POCKET6',(#347,#348));
#347=CARTESIAN_POINT('POLYLINE POINT 1',(19.0,-14.0,7.97));
#348=CARTESIAN_POINT('POLYLINE POINT 1',(19.0,19.0,7.97));
#349=TRIMMED_CURVE('TRIMMED CURVE FOR CONTOUR OF
POCKET6',#350,(#351),(#352),.T.,.CARTESIAN.);
#350=CIRCLE('CIRCLE',#353,6.0);

```

```

#351=CARTESIAN_POINT('TRIM POINT 1',(19.0,19.0,7.97));
#352=CARTESIAN_POINT('TRIM POINT 2',(13.0,25.0,7.97));
#353=AXIS2_PLACEMENT_3D('CIRCLE PLACEMENT',#354,#355,#356);
#354=CARTESIAN_POINT('CIRCLE CENTER',(13.0,19.0,7.97));
#355=DIRECTION('Z DIRECTION',(0.0,0.0,1.0));
#356=DIRECTION('X DIRECTION',(1.0,0.0,0.0));
#357=STEP('STEP1',#3,(#358),#359,#360,#361,#362,());
#358=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'STEP1',$,$,#363,#364,#365,$,#366,#367,#368,$,$,
$,$);
#359=AXIS2_PLACEMENT_3D('STEP1 PLACEMENT',#371,#372,#373);
#360=ELEMENTARY_SURFACE('STEP1 DEPTH PLANE',#374);
#361=LINEAR_PATH(#378,#379,#380);
#362=VEE_PROFILE(#384,#385,90.0,90.0);
#363=MILLING_CUTTING_TOOL('T5',#369,(),$,$,$,$);
#364=MILLING_TECHNOLOGY(0.021016666666666666,TCP,,$,65.68333333333334,$,$,$,$,$);
#365=MILLING_MACHINE_FUNCTIONS(.F.,$,$,$,$,(),$,$,$,());
#366=PLUNGE_RAMP($,$);
#367=PLUNGE_RAMP($,$);
#368=BIDIRECTIONAL($,$,$,$,$);
#369=ENDMILL(#370,$,$,$,$);
#370=MILLING_TOOL_DIMENSION(16.0,$,$,$,0.0,$,$);
#371=CARTESIAN_POINT('STEP1',(60.0,107.0,0.0));
#372=DIRECTION('AXIS',(0.0,0.0,1.0));
#373=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#374=AXIS2_PLACEMENT_3D('STEP1 DEPTH',#375,#376,#377);
#375=CARTESIAN_POINT('STEP1 DEPTH',(0.055999999999997385,0.6209999999999951,-6.0));
#376=DIRECTION('AXIS',(0.0,0.0,1.0));
#377=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#378=AXIS2_PLACEMENT_3D('LINEAR PATH PLACEMENT',#381,#382,#383);
#379=TOLERANCED_LENGTH_MEASURE(100.0,$);
#380=DIRECTION('STEP DIRECTION',(0.0,-1.0,0.0));
#381=CARTESIAN_POINT('LINEAR PATH LOCATION',(-22.997,-7.0,0.0));
#382=DIRECTION('AXIS',(0.0,0.0,1.0));
#383=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#384=AXIS2_PLACEMENT_3D('VEE PROFILE PLACEMENT',#386,#387,#388);
#385=TOLERANCED_LENGTH_MEASURE(0.0,$);
#386=CARTESIAN_POINT('VEE PROFILE LOCATION',(-22.997,-7.0,-6.0));
#387=DIRECTION('AXIS',(0.7071067811865476,0.0,0.7071067811865476));
#388=DIRECTION('AXIS',(0.7071067811865476,0.0,-0.7071067811865476));
#389=SLOT('SLOT1',#3,(#390),#391,#392,#393,#394,());
#390=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'SLOT1',$,$,#289,#395,#396,$,#397,#398,#399,$,$,
$,$);
#391=AXIS2_PLACEMENT_3D('SLOT1 PLACEMENT',#400,#401,#402);
#392=ELEMENTARY_SURFACE('SLOT1 DEPTH PLANE',#403);
#393=LINEAR_PATH(#407,#408,#409);
#394=SQUARE_U_PROFILE(#413,#414,#415,0.0,$,0.0);
#395=MILLING_TECHNOLOGY(0.021016666666666666,TCP,,$,131.36666666666667,$,$,$,$,$);
#396=MILLING_MACHINE_FUNCTIONS(.F.,$,$,$,$,(),$,$,$,());
#397=PLUNGE_ZIGZAG($,$,$);
#398=PLUNGE_TOOLAXIS($);
#399=CENTER_MILLING($,$);
#400=CARTESIAN_POINT('SLOT1',(44.0,72.0,-5.97));
#401=DIRECTION('AXIS',(0.0,0.0,1.0));
#402=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#403=AXIS2_PLACEMENT_3D('SLOT1 DEPTH',#404,#405,#406);

```



```
#404=CARTESIAN_POINT('SLOT1 DEPTH',(-8.0,0.5,-5.03));
#405=DIRECTION('AXIS',(0.0,0.0,1.0));
#406=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#407=AXIS2_PLACEMENT_3D('PLACEMENT OF TRAVEL COURSE: SLOT1',#410,#411,#412);
#408=TOLERANCED_LENGTH_MEASURE(15.0,$);
#409=DIRECTION('DIRECTION: SLOT1',(-1.0,0.0,0.0));
#410=CARTESIAN_POINT('POINTSLOT1',(7.0,0.5,-5.03));
#411=DIRECTION('AXIS',(0.0,0.0,1.0));
#412=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#413=AXIS2_PLACEMENT_3D('PLACEMENT OF SWEPT SHAPE: SLOT1',#416,#417,#418);
#414=TOLERANCED_LENGTH_MEASURE(8.0,$);
#415=TOLERANCED_LENGTH_MEASURE(0.0,$);
#416=CARTESIAN_POINT('POINTSLOT1',(51.0,72.5,-11.0));
#417=DIRECTION('AXIS',(0.0,0.0,1.0));
#418=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#419=AXIS2_PLACEMENT_3D('SETUP ORIGIN',#421,#422,#423);
#420=WORKPIECE_SETUP(#3,#424,$,$,());
#421=CARTESIAN_POINT('SETUP LOCATION',(0.0,0.0,0.0));
#422=DIRECTION('AXIS',(0.0,0.0,1.0));
#423=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
#424=AXIS2_PLACEMENT_3D('WORKPIECE50.0X100.0X20.0 SETUP',#425,#426,#427);
#425=CARTESIAN_POINT('WORKPIECE50.0X100.0X20.0SETUP LOCATION',(0.0,0.0,0.0));
#426=DIRECTION('AXIS',(0.0,0.0,1.0));
#427=DIRECTION('REF_DIRECTION',(1.0,0.0,0.0));
ENDSEC;

END-ISO-10303-21;
```


E_LAB_E_PLANAR_FACE3_BOSS:

E_LAB_A_POCKET2:

G17 G90 DIAMOF ; *GP*

G0 X15.0 Y90.0;*GP*

G3 X10.0 Y85.0 I=AC(15.0) J=AC(85.0);*GP*

G1 X10.0 Y75.0;*GP*

G3 X15.0 Y70.0 I=AC(15.0) J=AC(75.0);*GP*

G1 X20.0 Y70.0;*GP*

G2 X25.0 Y65.0 I=AC(20.0) J=AC(65.0);*GP*

G1 X25.0 Y40.0;*GP*

G3 X30.0 Y35.0 I=AC(30.0) J=AC(40.0);*GP*

G1 X35.0 Y35.0;*GP*

G3 X40.0 Y40.0 I=AC(35.0) J=AC(40.0);*GP*

G1 X40.0 Y85.0;*GP*

G3 X35.0 Y90.0 I=AC(35.0) J=AC(85.0);*GP*

G1 X15.0 Y90.0;*GP*

RET;*GP*

;CON,0,0.0000,8,8,MST:0,0,AX:X,Y,I,J;*GP*;*RO*;*HD*

;S,EX:15.0,EY:90.0;*GP*;*RO*;*HD*

;ACCW,DIA:0/235,EX:10.0,EY:85.0,RAD:5.0;*GP*;*RO*;*HD*

;LA,EX:10.0,EY:75.0;*GP*;*RO*;*HD*

;ACCW,DIA:0/35,EX:15.0,EY:70.0,RAD:5.0;*GP*;*RO*;*HD*

;LR,EX:20.0,EY:70.0;*GP*;*RO*;*HD*

;ACW,DIA:0/235,EX:25.0,EY:65.0,RAD:5.0;*GP*;*RO*;*HD*

;LA,EX:25.0,EY:40.0;*GP*;*RO*;*HD*

;ACCW,DIA:0/35,EX:30.0,EY:35.0,RAD:5.0;*GP*;*RO*;*HD*

;LR,EX:35.0,EY:35.0;*GP*;*RO*;*HD*

;ACCW,DIA:0/235,EX:40.0,EY:40.0,RAD:5.0;*GP*;*RO*;*HD*

;LU,EX:40.0,EY:85.0;*GP*;*RO*;*HD*

;ACCW,DIA:0/35,EX:35.0,EY:90.0,RAD:5.0;*GP*;*RO*;*HD*

;LL,EX:15.0,EY:90.0;*GP*;*RO*;*HD*

E_LAB_E_POCKET2:

C.3.3. Siemens part programme generated from STEP-NC file - Part III

```

N5 E_HEAD(268439551,0,0,0,50,100,-20,71,17,1,100,0,0,6);*RO*
N10
E_MI_PL("66mmFacemill","10mmSlot_c",1,1000,1,2000,1,9,0,90,1,91,15,0,0,90,0,90,50,90,100
,90,22,0);*RO*
N15 E_CON("POCKET1",1,"E_LAB_A_POCKET1","E_LAB_E_POCKET1");*RO*
N16 E_CP_CO("10mmSlot_c","16mmSlot",1,1261,1,5255,1,"2012052417332200",1,5,0,-
9,90,4,2,0,0,0,0,500,1,1,11,100,1,0,"00000703109244248698","00000056406048031072",1,1,
1,1,,"",0,5.);*RO*
N20 E_CON("POCKET2",1,"E_LAB_A_POCKET2","E_LAB_E_POCKET2");*RO*
N21 E_CP_CO("10mmSlot_c","16mmSlot",1,1262,1,5256,1,"2012052417332201",1,5,0,-
9,90,4,2,0,0,0,0,500,1,1,11,100,1,0,"00000703606741464429","00000056202029649418",0,1,
1,1,,"",0,5.);*RO*
N25 E_CON("POCKET4",1,"E_LAB_A_POCKET4","E_LAB_E_POCKET4");*RO*
N26 E_CP_CO("10mmSlot_c","16mmSlot",1,1261,1,6306,1,"2012052416580202",1,5,0,-
9,90,3,3333,2,0,5,1,25,0,0,150,1,1,11,100,1,0,"00000705401049381781","00000056409731817
619",0,1,1,1,,"",0,5.);*RO*
N65 E_CP_CO("10mmSlot_c","16mmSlot",1,1164,1,9702,1,"2012052416580203",4,5,0,-
9,90,3,3333,2,0,5,0,0,0,15,3,0,100,1,0,"00000572206691003503","00000056404021965352",0,
1,1,1,,"",0,6,5);*RO*
N30 E_CP_CO("10mmSlot_c","16mmSlot",1,1164,1,9702,1,"2012052416580304",2,5,0,-
9,90,3,3333,15,0,5,2,0,0,0,150,1,0,100,1,0,"00000676202347700416","00000056403958278622"
,0,1,1,1,,"",0,5.);*RO*
N35 E_CON("POCKET6",1,"E_LAB_A_POCKET6","E_LAB_E_POCKET6");*RO*
N36 E_CP_CO("10mmSlot_c","16mmSlot",1,1261,1,6306,1,"2012052416580305",1,5,0,-
9,90,3,3333,2,1,1,0,0,0,150,1,1,11,100,1,0,"00000706809243438519","00000056502496980369"
,0,1,1,1,,"",0,14.8999);*RO*
N70 E_CP_CO("10mmSlot_c","16mmSlot",1,864,1,9000,1,"2012052416580306",4,5,0,-
9,90,2,6667,2,1,0,0,0,0,15,3,0,100,1,0,"00000573502525088688","00000056504008142686",0,1
,1,1,,"",0,7.);*RO*
N40 E_CP_CO("10mmSlot_c","16mmSlot",1,864,1,9000,1,"2012052416580407",2,5,0,-
9,90,2,15,1,1,0,0,0,150,1,0,100,1,0,"00000677005895424885","00000055901244286219",0,1,1,
1,,"",0,16.88);*RO*
N45 E_CON("STEP1",1,"E_LAB_A_STEP1","E_LAB_E_STEP1");*RO*
N46 E_CP_CO("16mmSlot","8mmSlot",1,1261,1,3941,1,"2012052416580408",1,8,0,-
6,90,5,3333,2,0,0,0,0,0,150,1,1,11,100,1,0,"00000602309535325842","00000057700659447367"
,0,1,1,1,,"",0,13.28);*RO*
N50 E_SL_LON(4,0,0,"8mmSlot",,"",1,1261,1,7882,1,1,150,1,0,90,-
11,90,2,0,43.5,90,72.5,90,23,8,0,0,0,2.6667,0,0);*RO*
N55 M30 ;#SM;*RO*

E_LAB_A_POCKET1:
G17 G90 DIAMOF ; *GP*
G0 X27.0 Y25.0;*GP*
G1 X23.0 Y25.0;*GP*
G3 X17.0 Y19.0 I=AC(23.0) J=AC(19.0);*GP*
G1 X17.0 Y10.0;*GP*
G3 X23.0 Y4.0 I=AC(23.0) J=AC(10.0);*GP*
G1 X27.0 Y4.0;*GP*
G3 X33.0 Y10.0 I=AC(27.0) J=AC(10.0);*GP*
G1 X33.0 Y19.0;*GP*
G3 X27.0 Y25.0 I=AC(27.0) J=AC(19.0);*GP*
RET ;*GP*

```

```
;CON,0,0.0000,8,8,MST:0,0,AX:X,Y,I,J;*GP*;*RO*;*HD*
;S,EX:27.0,EY:25.0;*GP*;*RO*;*HD*
;LL,EX:23.0,EY:25.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/235,EX:17.0,EY:19.0,RAD:6.0;*GP*;*RO*;*HD*
;LA,EX:17.0,EY:10.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/35,EX:23.0,EY:4.0,RAD:6.0;*GP*;*RO*;*HD*
;LR,EX:27.0,EY:4.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/235,EX:33.0,EY:10.0,RAD:6.0;*GP*;*RO*;*HD*
;LU,EX:33.0,EY:19.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/35,EX:27.0,EY:25.0,RAD:6.0;*GP*;*RO*;*HD*
E_LAB_E_POCKET1:
```

```
E_LAB_A_POCKET2:
G17 G90 DIAMOF ; *GP*
G0 X27.0 Y46.0;*GP*
G1 X23.0 Y46.0;*GP*
G3 X17.0 Y40.0 I=AC(23.0) J=AC(40.0);*GP*
G1 X17.0 Y31.0;*GP*
G3 X23.0 Y25.0 I=AC(23.0) J=AC(31.0);*GP*
G1 X27.0 Y25.0;*GP*
G3 X33.0 Y31.0 I=AC(27.0) J=AC(31.0);*GP*
G1 X33.0 Y40.0;*GP*
G3 X27.0 Y46.0 I=AC(27.0) J=AC(40.0);*GP*
RET ;*GP*
;CON,0,0.0000,8,8,MST:0,0,AX:X,Y,I,J;*GP*;*RO*;*HD*
;S,EX:27.0,EY:46.0;*GP*;*RO*;*HD*
;LL,EX:23.0,EY:46.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/235,EX:17.0,EY:40.0,RAD:6.0;*GP*;*RO*;*HD*
;LA,EX:17.0,EY:31.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/35,EX:23.0,EY:25.0,RAD:6.0;*GP*;*RO*;*HD*
;LR,EX:27.0,EY:25.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/235,EX:33.0,EY:31.0,RAD:6.0;*GP*;*RO*;*HD*
;LU,EX:33.0,EY:40.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/35,EX:27.0,EY:46.0,RAD:6.0;*GP*;*RO*;*HD*
E_LAB_E_POCKET2:
```

```
E_LAB_A_POCKET4:
G17 G90 DIAMOF ; *GP*
G0 X41.0 Y33.0;*GP*
G1 X9.0 Y33.0;*GP*
G3 X4.0 Y28.0 I=AC(9.0) J=AC(28.0);*GP*
G1 X4.0 Y22.0;*GP*
G3 X9.0 Y17.0 I=AC(9.0) J=AC(22.0);*GP*
G1 X41.0 Y17.0;*GP*
G3 X46.0 Y22.0 I=AC(41.0) J=AC(22.0);*GP*
G1 X46.0 Y28.0;*GP*
G3 X41.0 Y33.0 I=AC(41.0) J=AC(28.0);*GP*
RET ;*GP*
;CON,0,0.0000,8,8,MST:0,0,AX:X,Y,I,J;*GP*;*RO*;*HD*
;S,EX:41.0,EY:33.0;*GP*;*RO*;*HD*
;LL,EX:9.0,EY:33.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/235,EX:4.0,EY:28.0,RAD:5.0;*GP*;*RO*;*HD*
;LA,EX:4.0,EY:22.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/35,EX:9.0,EY:17.0,RAD:5.0;*GP*;*RO*;*HD*
;LR,EX:41.0,EY:17.0;*GP*;*RO*;*HD*
```

```
;ACCW,DIA:0/235,EX:46.0,EY:22.0,RAD:5.0;*GP*;*RO*;*HD*
;LU,EX:46.0,EY:28.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/35,EX:41.0,EY:33.0,RAD:5.0;*GP*;*RO*;*HD*
E_LAB_E_POCKET4:
```

```
E_LAB_A_POCKET6:
G17 G90 DIAMOF ; *GP*
G0 X37.0 Y95.0;*GP*
G1 X13.0 Y95.0;*GP*
G3 X7.0 Y89.0 I=AC(13.0) J=AC(89.0);*GP*
G1 X7.0 Y56.0;*GP*
G3 X13.0 Y50.0 I=AC(13.0) J=AC(56.0);*GP*
G1 X37.0 Y50.0;*GP*
G3 X43.0 Y56.0 I=AC(37.0) J=AC(56.0);*GP*
G1 X43.0 Y89.0;*GP*
G3 X37.0 Y95.0 I=AC(37.0) J=AC(89.0);*GP*
RET ;*GP*
;CON,0,0.0000,8,8,MST:0,0,AX:X,Y,I,J;*GP*;*RO*;*HD*
;S,EX:37.0,EY:95.0;*GP*;*RO*;*HD*
;LL,EX:13.0,EY:95.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/235,EX:7.0,EY:89.0,RAD:6.0;*GP*;*RO*;*HD*
;LA,EX:7.0,EY:56.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/35,EX:13.0,EY:50.0,RAD:6.0;*GP*;*RO*;*HD*
;LR,EX:37.0,EY:50.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/235,EX:43.0,EY:56.0,RAD:6.0;*GP*;*RO*;*HD*
;LU,EX:43.0,EY:89.0;*GP*;*RO*;*HD*
;ACCW,DIA:0/35,EX:37.0,EY:95.0,RAD:6.0;*GP*;*RO*;*HD*
E_LAB_E_POCKET6:
```

```
E_LAB_A_STEP1:
G17 G90 DIAMOF ; *GP*
G0 X37.00299835205078 Y116.0;*GP*
G1 X37.00299835205078 Y-16.0;*GP*
G1 X66.0 Y-16.0;*GP*
G1 X66.0 Y116.0;*GP*
G1 X37.00299835205078 Y116.0;*GP*
RET ;*GP*
;CON,0,0.0000,8,8,MST:0,0,AX:X,Y,I,J;*GP*;*RO*;*HD*
;S,EX:37.00299835205078,EY:116.0;*GP*;*RO*;*HD*
;LA,EX:37.00299835205078,EY:-16.0;*GP*;*RO*;*HD*
;LR,EX:66.0,EY:-16.0;*GP*;*RO*;*HD*
;LU,EX:66.0,EY:116.0;*GP*;*RO*;*HD*
;LL,EX:37.00299835205078,EY:116.0;*GP*;*RO*;*HD*
E_LAB_E_STEP1:
```